

Advanced Desktop Computer Series

Technical Reference Manual

Printed in U.S.A

595-3922

ZENITH | data
systems

THE QUALITY GOES IN BEFORE THE NAME GOES ON

FEDERAL COMMUNICATIONS COMMISSION RADIO FREQUENCY INTERFERENCE STATEMENT

WARNING: As sold by the manufacturer, the equipment described in this manual has been certified to comply with the limits for a Class B computing device, pursuant to Subpart J of Part 15 of FCC rules. Only peripherals (computer input/output devices, terminals, printers, etc.) certified to comply with the Class B limits may be attached to this computer. Operation with non-certified peripherals is likely to result in interference with radio and TV reception.

NOTE: In order to meet Class B emission limits, the I/O cables that interconnect between the computer and any peripheral (such as a printer, external modem, etc.) must be shielded.

USER MODIFICATIONS

The user is responsible for any interference with radio or TV reception caused by a user-modified computer. The manufacturer supplied this manual for the sole purpose of providing technical information concerning its product(s). The user is responsible for any user-modifications made to the software, firmware, or hardware that are the result of information supplied in this manual.

The manufacturer reserves the right to make changes at any time to the design of the product(s) without obligation to update existing product(s), including the information in this manual.

Limited Rights Legend

Contractor is Zenith Data Systems Corporation of St. Joseph, Michigan 49085.
The entire document is subject to Limited Rights data provisions.

ACKNOWLEDGEMENTS

We wish to thank Intel Corporation for granting us permission to reprint technical information and illustrations used in the publication of this manual.

Copyright 1987 by Zenith Data Systems Corporation.
Printed in the United States of America.

Zenith Data Systems Corporation
St. Joseph, Michigan 49085

Preface

This technical manual is written with the advanced user in mind. Every effort has been made to provide the most information in the least amount of space. To this end the technical material contained herein is as descriptive and as concise as possible. Our goal is to provide the advanced, technically-oriented user and/or programmer with as much factual, technically accurate material as we possibly could. Because of this, many individuals who lack a sufficient technical background may find this material difficult to grasp.

A large number of well-written books are available through commercial outlets that may help bridge this background gap. These books, by many different authors, are available from numerous publishing firms covering the entire technical range from computer basics to advanced systems development. We highly recommend that anyone who feels the need, review these books as possible additions to their reference material. In addition, we have identified several specific sources of technical information related to the material within this manual. Most of these are manufacturer-specific publications relating to a specific product or line of products. In most cases, like this manual, they are of a highly technical nature. If the user needs basic, introductory technical material, we recommend the commercially available sources mentioned above.



Contents

Part I — Introduction

Chapter 1 — Introduction and General Information

Manual Description	1-2
Related Publications	1-2
Product Applications	1-4
Product Information	1-5
Environment	1-5
Optional Cards	1-6
Coprocessors	1-7
Video Options	1-7
Mass Storage Options	1-7
Specifications	1-9

Chapter 2 — Installation

Required Tools	2-1
Operating Environment and Power Requirements	2-1
Unpacking and Setting Up	2-2
Connecting Peripherals	2-3
Initial Power-up	2-8
Self-Testing	2-8
Autobooting	2-8
Entering the Monitor Program	2-9
Booting Manually	2-10
Loading the Operating System	2-10
Resetting the Computer	2-11
Installing Internal Options	2-11
Installing a Circuit Card	2-12
Installing an Integrated Circuit	2-14

Chapter 3 — Disassembly

Cover Removal	3-1
Circuit Card Removal	3-2
Auxiliary Fan Removal	3-3
Backplane Board Removal	3-4
Power Supply Removal	3-9
Disk Drive Removal	3-11

Contents

Chapter 4 — Hardware Configuration

CPU Card	4-1
I/O Card	4-4
Memory	4-7
System Cache Card	4-11
Floppy/Winchester Controller Card	4-11
Disk Drives	4-12
Floppy Drives	4-12
Fixed Hard Disk Drives	4-20
Removable Cartridge Hard Disk Drives	4-22
Video System	4-24
Video Drivers	4-27

Chapter 5 — Operation

Keyboard	5-1
AT/XT Compatibility	5-8
Keyboard Adjustment	5-9
Using the Setup/Configuration Program	5-9
System Boot Procedure	5-14

PART II — System Programming

Chapter 6 — The Monitor Program and Programming with Interrupts

The Monitor Program	6-1
Autoboot	6-2
User Commands	6-4
Video Commands and Disk Boot Routines	6-5
Help	6-5
Display Color Bar	6-5
Set Video/Scroll Mode	6-6
Boot Disk	6-7
The Machine Language Debugger	6-8
Display Memory	6-9
Examine Memory	6-10
Fill Memory	6-10
Go (Execute)	6-11
Hex Math	6-12
Input From Port	6-12
Move Memory Block	6-12
Output To Port	6-13
Examine/Modify Registers	6-13
Search Memory	6-15
Trace User Program	6-15
Unassemble	6-16

Programming with Interrupts	6-17
Hardware Interrupts	6-17
Using a Software Interrupt	6-19
Modifying an Interrupt	6-20
Software Interrupt Summary	6-21
System Organization	6-22
Monitor Program Jump Vectors	6-24

Chapter 7 — System and CPU Interrupts

Programming System and CPU Interrupts	7-1
Divide by Zero (INT 00H)	7-1
Single Step (INT 01H)	7-2
Non-maskable Interrupt (INT 02H)	7-2
Software Breakpoint (INT 03H)	7-2
Arithmetic Overflow (INT 04H)	7-3
Timer (Time-of-Day) (INT 08H)	7-3
Real-Time Clock (INT 0AH)	7-3
Equipment Configuration (INT 11H)	7-4
Memory Size (INT 12H)	7-5
Device Control (INT 15H)	7-5
Set/Read the Time of Day (INT 1AH)	7-14
Tick Timer (INT 1CH)	7-17
Alarm Interrupt (INT 4AH)	7-17
Real-Time Clock Alarm Interrupt (70H)	7-18
Programming Sound	7-18

Chapter 8 — Keyboard

Programming Keyboard Interrupts	8-1
Key Pressed (INT 09H)	8-1
Keyboard Input/Output (INT 16H)	8-2
Keyboard Break (INT 1BH)	8-7
Keyboard Codes	8-8
System Scan Codes	8-17
Alphabetic Keys	8-17
Numeric and Punctuation Keys	8-19
Function and Control Keys	8-20
Make/Break Key Codes	8-25

Contents

Chapter 9 — Input/Output Interrupts

Programming Input/Output Interrupts	9-1
Print Screen (INT 05H)	9-1
Communications (INT 0BH and INT 0CH)	9-2
Parallel Printer (INT 0DH and INT 0FH)	9-2
Serial Input/Output (INT 14H)	9-2
Printer Input/Output (INT 17H)	9-7
Parallel/Serial Configuration (INT 18H)	9-8
Parallel Format	9-9
Serial Format	9-10

Chapter 10 — Disk Drive Interrupts

Floppy and Hard Disk Drive Hardware Interrupt Return (INT 0EH and INT 76H)	10-1
Disk Input/Output (INT 13H and INT 40H)	10-2
Drive Selection	10-5
Error Status Codes	10-6
Function Call Codes	10-7
Bootting an Operating System (INT 19H)	10-17
Floppy Disk Parameters (INT 1EH)	10-18
Hard Disk Parameters (INT 41H, INT 46H, and INT 4BH) ...	10-24

Chapter 11 — Video Interrupts

Programming the Video Interrupts	11-1
Video Input/Output (INT 10H)	11-1
Video Initialization (INT 1DH)	11-12
Defining Characters (INT 1FH)	11-13
EGA Video Considerations	11-14
Monochrome Video Modes	11-15
Normal Color Video Modes	11-15
Enhanced Color Video Modes	11-17
Basic Modes of Operation	11-17
Text (Alphanumeric) Modes	11-18
Graphics Modes	11-20
Medium-Resolution Color	11-20
High-Resolution Color	11-21
Mode F	11-21
Mode 10	11-23
Register Default Values	11-24

Part III — System Hardware

Chapter 12 — The CPU

Introduction	12-1
80386 Base Architecture	12-2
Register Resources	12-4
General Purpose Registers	12-9
Instruction Pointer and Flags	12-9
Segment Registers	12-11
Control Registers	12-12
Address Registers	12-14
Debug and Test Registers	12-16
Processor Addressing	12-17
Memory	12-18
Input/Output	12-19
Interrupts	12-19
Debug Registers	12-22
Test Registers	12-26
Real Mode	12-28
Protected Architecture	12-29
General Protection Concepts	12-29
Descriptors	12-29
Segment Access	12-31
Privilege	12-31
Privilege Types	12-32
Inter-Segment Privilege Transfers	12-33
Call Gates	12-34
Task Switching	12-35
Paging	12-38
Translation Lookaside Buffer	12-40
Descriptor Attributes	12-42
Code and Data Descriptors	12-44
System Segment Descriptors	12-46
Memory Addressing	12-49
Protected Mode Initialization	12-50
Virtual 8086 Mode	12-52
Virtual Mode Addressing	12-52
Paging	12-52
Protection and I/O Permission	12-53
Interrupts	12-55
Virtual Mode Transitions	12-56
80386 Pinouts	12-58

Contents

Signal Descriptions	12-62
Bus Signals	12-62
Bus Arbitration	12-64
Bus Cycle Definition	12-64
Bus Control	12-65
Coprocessor Interface	12-66
Interrupt Signals	12-66
Miscellaneous	12-67

Chapter 13 — Coprocessors

80287	13-1
Architecture	13-2
Bus Interface Unit	13-3
Numeric Execution Unit	13-7
Register Resources	13-7
Operation	13-10
Programming	13-10
Device Pinout	13-11
80387	13-13
Architecture	13-13
Bus Control Unit	13-14
Data Interface/Control Unit	13-14
Floating Point Unit	13-18
Register Resources	13-19
Operation	13-21
Programming	13-22
Device Pinout	13-23
80387 Pin Descriptions	13-25
Control Signals	13-25
Handshake Signals	13-25
Bus Interface Signals	13-26
Chip Select Signals	13-27
Power Signals	13-27

Chapter 14 — Support Circuits

CPU Support Circuitry	14-1
The 8259 Interrupt Controller	14-1
Interrupt Controller Architecture	14-3
Initialization	14-5
Operation	14-9
Interrupt Controller Port Address	14-16
Programming Considerations	14-16
Sequence of Operation	14-17
Further Programming	14-18
Pinout	14-20

The 8254 Programmable Interval Timer	14-23
Mode Definitions	14-25
Programming Considerations	14-30
Pinout	14-32
8237 DMA Controller	14-34
Internal Registers	14-35
Operation	14-42
Special Features	14-45
Programming Considerations	14-47
Pinout	14-47
The MC146818A Real-Time Clock	14-51
Architecture	14-51
Internal Registers	14-52
Register A	14-54
Register B	14-55
Register C	14-56
Register D	14-56
Time, Calendar, and Alarm Registers	14-56
Operation	14-57
Programming	14-58
Initialization	14-58
Interrupts	14-59
Update Cycle	14-60
Memory	14-61
Pinout	14-62
System Control Processor	14-64
The Status Register	14-65
Buffers	14-66
SCP Commands	14-67
Keyboard Operations	14-70
Receiving Keyboard Data	14-70
Sending Keyboard Data	14-71
Slushware	14-71
CPU Protected Mode	14-72
SCP Pinout	14-73
Special Programming Features	14-75
Fast GATEA20	14-76
Fast CPU Reset	14-76
Scratchpad RAM Enable	14-76
NMI Enable	14-77
 Chapter 15 — Memory	
System Memory	15-3
System Memory Addressing	15-6
User Memory	15-6
Memory Address Format	15-7

Contents

Address Decoding	15-9
Data Bus Interface	15-9
Paging Controller	15-12
Refresh (RAS/CAS)	15-12
Parity	15-13
Memory Bank Configuration	15-15
Slushware	15-16
EMS Memory	15-17
Cache Memory	15-21
Main Memory to Cache Memory Mapping	15-23
Memory Read	15-23
Memory Write	15-26
Dynamic Memory Control	15-26
 Chapter 16 — Mass Storage	
Supported Drives	16-1
Drive Selection	16-2
Error Detection and Invalid Commands	16-2
Floppy Disk Control	16-3
DMA and Non-DMA Modes	16-3
Master Control Logic	16-4
Serial Controller	16-4
Drive Controller	16-5
FDC Control Registers	16-5
Digital Output Register (Port 3F2H)	16-6
Main Status Register (Port 3F4H)	16-7
Floppy Control Register (Port 3F7H)	16-9
Data Register (Port 3F5H)	16-9
FDC Pinout	16-13
Programming Floppy Disk Operations	16-17
Floppy Disk Controller Commands	16-18
Hard Disk Control	16-36
Hard Disk Drive Control Registers	16-37
Data Register (Port 1F0H)	16-37
Error Register (Port 1F1H)	16-38
Write-Precompensation Register (Port 1F1H)	16-40
Sector Count Register (Port 1F2H)	16-40
Sector Number Register (Port 1F3H)	16-40
Cylinder Low and Cylinder High Registers (Ports 1F4H and 1F5H)	16-40
Drive and Head Register (Port 1F6H)	16-41
Status Register (Port 1F7H)	16-41
Command Register (Port 1F7H)	16-43
Fixed Disk Register (Port 3F6H)	16-45
Digital Input Register (Port 3F7H)	16-45
Programming Hard Disk Operations	16-46
Hard Disk Controller Commands	16-46

Chapter 17 — 31 kHz Video

Video Systems Overview	17-1
Raster-Scan Technology	17-1
CRT Controller Concepts	17-7
Controller IC	17-8
Video Memory	17-8
Shift Registers and the Video Encoder	17-9
Hardware Features	17-9
Basic Video Modes	17-10
Text (Alphanumeric) Modes	17-10
Video Systems	17-17
MDA System	17-17
HGC System	17-17
CGA System	17-18
EGA System	17-18
VGA System	17-19
PGA System	17-19
31 kHz Video Card	17-19
6845 CRT Controller Emulation	17-20
Pointer Address Register	17-21
Data Registers	17-22
Video Mode Select Register	17-26
Color Select Register	17-27
Status Register	17-29
Light Pen Register	17-30
MDA Interface	17-30
Control Register	17-30
Status Register	17-36
CGA Interface	17-37
CRT Controller Registers	17-37
Programming Sequence	17-49
HGC Interface	17-50
Index and Data Registers	17-50
Display Mode Control Register	17-50
Display Status Register	17-52
Configuration Register	17-53
Enhanced Graphics	17-56
82C431 Graphics Controller	17-58
82C431 Pinout	17-68
82C432 Sequencer	17-73
82C432 Pinout	17-79
82C433 Attributes Controller	17-83
82C433 Pinout	17-89
82C434 CRT Controller	17-95
82C434 Pinout	17-110
31 KHz Interface	17-118
31 kHz Extended Chip Set	17-118
Color Palette	17-118

Contents

Chapter 18 — Serial and Parallel Communications

Serial Port	18-1
NS16450 Asynchronous Communications Element	18-2
Handshaking	18-4
NS16450 Programming	18-4
Pinout	18-11
Parallel Port	18-16

Chapter 19 — ROM-Based Tests and Error Messages

Self-Tests	19-1
Selectable Tests	19-2
Test Menu	19-2
The Disk Read Test	19-3
The Keyboard Test	19-4
The Base Memory Test	19-4
The Expansion Memory Test	19-5
The Power-up Test	19-5
Exiting the Test Menu	19-5
Error Messages	19-6
Error Messages Related to Disk Drives	19-6
Error Messages Related to The Setup/Configuration Program	19-8
General Error Messages	19-9
Error Summary Lines	19-11
Disk-Based Diagnostics	19-12

Appendix A — 80386 Instruction Set

Appendix B — 80287 Instruction Set

Appendix C — 80387 Instruction Set

Figures

1-1.	Advanced Desktop Computer	1-1
2-1.	The Back Panel of the Computer	2-2
2-2.	The Video Connectors	2-5
2-3.	Connecting the Video Monitor	2-6
2-4.	The Keyboard Connector	2-6
2-5.	The Serial/Parallel Connectors	2-7
2-6.	Standard and Optional Card Locations	2-13
2-7.	Installing Optional ICs	2-15
3-1.	Cover Removal	3-2
3-2.	Circuit Card Removal	3-3
3-3.	Auxiliary Fan Removal	3-4
3-4.	Power Supply Connection	3-5
3-5.	Power Supply Ground Lead	3-6
3-6.	Backplane Board Removal	3-7
3-7.	Spring and Retainer	3-8
3-8.	Drive Power Connections	3-9
3-9.	Power Supply Removal	3-10
3-10.	Disk Drive Removal	3-12
4-1.	CPU Switch and Jumper Positions	4-1
4-2.	Memory Card Base Address Selection	4-2
4-3.	I/O Card Configuration Jumpers	4-5
4-4.	P301 Connector Signals	4-6
4-5.	System Memory Card	4-7
4-6.	SW401 Default Settings	4-7
4-7A.	Memory Configuration Options	4-8
4-7B.	Memory Configuration Options	4-9
4-8.	Drive Controller Card	4-11
4-9.	Optional Drive Modification	4-13
4-10.	5.25-Inch 360K Drive	4-14
4-11.	5.25-Inch 1.2M Drive	4-15
4-12.	3.25-Inch 720K Drive	4-16
4-13.	Drive Type A Configuration Settings	4-17
4-14.	Drive Type B Configuration Settings	4-18
4-15.	Drive Type C Configuration Settings	4-18
4-16.	Drive Type D Configuration Settings	4-19
4-17.	5.25-Inch 20M Drive	4-20
4-18.	5.25-Inch 20M/40M Drive	4-21
4-19.	5.25-Inch 40M/80M Drive	4-22
4-20.	Removable Cartridge Drive	4-23
4-21.	31 kHz Video Switches and Jumpers	4-24

Contents

5-1.	101-Key Keyboard	5-1
5-2.	Alphanumeric Mode	5-2
5-3.	Screen Control and Calculator Keypad Keys	5-5
5-4.	Function Keys	5-7
5-5.	AT/XT Compatibility Switch	5-8
5-6.	Keyboard Adjustment	5-9
5-7.	Setup Menu	5-10
6-1.	MFM-300 Command Summary Menu	6-4
7-1.	Block Move Descriptor Table	7-8
7-2.	Protected Mode Descriptor Table	7-10
8-1.	101-Key keyboard	8-16
8-2.	Alphabetic Keys	8-16
8-3.	Numeric and Punctuation keys	8-19
8-4.	Function and Control Keys	8-20
9-1.	Serial and Parallel Device Layout	9-8
11-1.	Character Design Matrix	11-14
12-1.	CPU Card Block Diagram	12-1
12-2.	80386 Processor Architecture	12-3
12-3.	80386 Internal Register Structure	12-5
12-4.	80386 Flags Register	12-9
12-5.	80386 Control Registers	12-12
12-6.	80386 System Segment Registers	12-15
12-7.	80386 Debug and Test Registers	12-16
12-8.	80386 Debug Registers	12-23
12-9.	80386 Test Registers	12-26
12-10.	80386 TSS and TSS Registers	12-36
12-11.	Page Directory Entry Format	12-38
12-12.	Segment Descriptor Format	12-43
12-13.	Code and Data Descriptor Format	12-44
12-14.	System Segment Descriptor Format	12-46
12-15.	Gate Descriptor Format	12-48
12-16.	Protected Mode Addressing	12-49
12-17.	Protected System Model	12-50
12-18.	GDT Descriptors	12-51
12-19.	I/O Permission Bit Map	12-54
12-20.	80386 Pin Assignments	12-59
12-21.	Processor Timing Relationships	12-67

Contents

13-1.	80287 Block Diagram	13-2
13-2.	80287 Status Word Format	13-3
13-3.	80287 Control Word Format	13-6
13-4.	80287 Registers	13-8
13-5.	Register Mode Differences	13-9
13-6.	80287 Signal Pinout	13-11
13-7.	80387 Block Diagram	13-13
13-8.	80387 Status Word Format	13-15
13-9.	80387 Registers	13-19
13-10.	32-Bit Format Register Differences	13-20
13-11.	16-Bit Format Register Differences	13-21
13-12.	80387 Device Pinout	13-23
14-1.	8259 Interrupt Controller Block Diagram	14-3
14-2.	8259 Interrupt Controller Pinout	14-22
14-3.	8254 Programmable Interval Timer Block Diagram	14-23
14-4.	Timing Mode 0	14-25
14-5.	Timing Mode 1	14-26
14-6.	Timing Mode 2	14-27
14-7.	Timing Mode 3	14-28
14-8.	Timing Mode 4	14-29
14-9.	Timing Mode 5	14-29
14-10.	8254 Programmable Interval Timer Pinout	14-33
14-11.	8237 DMA Controller Block Diagram	14-34
14-12.	8237 DMA Controller Pinout	14-50
14-13.	MC146818A Real-Time Clock	14-52
14-14.	Internal Address Map	14-53
14-15.	MC146818A Status Registers	14-53
14-16.	Real-Time Clock Pinout	14-64
14-17.	Keyboard Data Format	14-70
14-18.	SCP Pinout	14-75
15-1.	Memory Map	15-5
15-2.	80386 Address Translation Process	15-8
15-3.	Memory Data Bus	15-11
15-4.	Conventional Memory	15-18
15-5.	Expanded Memory	15-19
15-6.	Cache Memory Block Diagram	15-21
15-7.	Cache/Main Memory Arbitration Logic	15-24
15-8.	Cache Read Miss Logic	15-25
16-1.	765 Disk Controller Block Diagram	16-4
16-2.	765 FDC Pinout	16-13

Contents

17-1.	Display Signal Relationships	17-4
17-2.	Scanning the Face of the CRT	17-5
17-3.	Horizontal Timing and Format	17-40
17-4.	82C431 Graphics Controller Pinout	17-72
17-5.	Video Memory Configuration	17-76
17-6.	82C432 Sequencer Pinout	17-83
17-7.	82C433 Attributes Controller Pinout	17-95
17-8.	Display Timing	17-96
17-9.	Soft Scrolling	17-97
17-10.	Cursor Control	17-98
17-11.	82C434 CRT Controller Pinout	17-117

18-1.	NS16450 Block Diagram	18-3
18-2.	NS16450 Pinout	18-15
18-3.	Parallel Connector	18-16

19-1.	Test Menu	19-3
-------	-----------------	------

Tables

2-1.	Video Controller Card Configuration	2-3
2-2.	Card Slot Assignments	2-13
2-3.	Optional IC Locations	2-15

4-1.	CPU Card Jumpers	4-3
4-2.	I/O Card Jumpers	4-6
4-3.	Video Option Selections	4-19

5-1.	Hard Disk Drive Types	5-12
------	-----------------------------	------

6-1.	Video and Disk Commands	6-5
6-2.	Video and Scroll Modes	6-6
6-3.	Machine Language Debugger Commands	6-8
6-4.	Processor Status Flag Codes	6-14
6-5.	Hardware Generated Interrupt Requests	6-18
6-6.	Interrupt Summary	6-21
6-7.	System Memory Map	6-22
6-8.	System Port Map	6-23

7-1.	System and CPU Interrupts	7-1
7-2.	Register AX Report from INT 11H	7-4
7-3.	Bits 4 and 5: Video Initialization	7-5
7-4.	Bits 6 and 7: Disk Drive Count	7-5
7-5.	INT 15 Functions	7-6
7-6.	Block Move Descriptor Table	7-9

Contents

7-7.	Protected Mode Table Descriptor	7-10
7-8.	Device Type Codes	7-12
7-9.	INT 15 Extended Functions	7-12
7-10.	INT 1AH Functions	7-14
8-1.	Keyboard Interrupts	8-1
8-2.	INT 16H Functions	8-2
8-3.	Keyboard Status Report	8-4
8-4.	Keyboard Status (0040:0018)	8-4
8-5.	Repetition Rates	8-5
8-6.	Keyboard Status Report (Register AL)	8-7
8-7.	Keyboard Status Report (Register AH)	8-7
8-8.	Alphabetic Keycode Sets	8-9
8-9.	Numeric and Punctuation Keycode Sets	8-10
8-10.	Function and Control Keycode Sets	8-11
8-11.	Factored Screen Control Keycode Sets	8-12
8-12.	Numeric Keycode Sets	8-15
8-13.	Alphabetic Key Codes (AT Mode)	8-15
8-14.	Numeric and Punctuation Key Codes	8-20
8-15.	Function and Control Keys	8-21
8-16.	Function and Control Key Scan Codes	8-24
8-17.	Make/Break Key Codes	8-26
9-1.	Input/Output Interrupts	9-1
9-2.	Serial Input/Output Function Codes	9-3
9-3.	Mode-Select Byte Breakdown	9-3
9-4.	Word Length Selection	9-4
9-5.	Parity Selection	9-4
9-6.	Baud Rate Selection	9-4
9-7.	Line Control Status	9-6
9-8.	Modem Control Status	9-6
9-9.	Printer Input/Output Function Codes	9-7
9-10.	Parallel Printer Status Report	9-7
9-11.	Parallel Map Format	9-9
9-12.	Parallel Map Byte #1	9-9
9-13.	Serial Map Format	9-10
9-14.	Byte #1 (Handshaking)	9-10
9-15.	Byte #2 (Parity and Case Mapping)	9-11
9-16.	Byte #7 (Word Length, Stop Bits, Parity, and Baud Rate)	9-12
10-1.	Disk Drive Interrupts	10-1
10-2.	Disk Drive Function Codes	10-2
10-3.	Drive Identification Codes	10-5
10-4.	INT 13H and INT 40H Error Status Codes	10-6

Contents

10-5.	CPU Register Initialization — Function Codes 02H – 04H and Function Codes 0AH – 0BH	10-8
10-6.	CPU Register Initialization — Function Code 05H	10-10
10-7.	CPU Register Results — Function Code 08H	10-11
10-8.	CPU Register Initialization — Function Code 0CH	10-13
10-9.	CPU Register Results — Function Code 0FH	10-14
10-10.	CPU Register Results — Function Code 10	10-14
10-11.	CPU Register Initialization — Function Code 11H	10-15
10-12.	CPU Register Results — Function Code 12H	10-16
10-13.	ROM Disk Parameters Table	10-18
10-14.	Drive Parameters — INT 1EH	10-21
10-15.	Floppy Disk Parameter Table	10-22
10-16.	Hard Disk Parameter Table	10-24
11-1.	Video Interrupts	11-1
11-2.	Video Input/Output Function Codes	11-2
11-3.	Video Modes	11-2
11-4.	Palette and Pixel Colors	11-6
11-5.	Row Specifier Options	11-10
11-6.	Register Values to Return Information for Function Code 11H	11-10
11-7.	EGA Information Returned by Function Code 12H	11-11
11-8.	Register, String, and Cursor Data for Function Code 13H	11-11
11-9.	Video Initialization Default Values	11-12
11-10.	Monochrome Video Modes	11-15
11-11.	Normal Color Video Modes	11-15
11-12.	Enhanced Color Video Modes	11-17
11-13.	Monochrome Attribute Byte Characteristics	11-19
11-14.	Color Attribute Byte Characteristics	11-19
11-15.	Color Selection	11-20
11-16.	Palette Colors	11-21
11-17.	Mode F Attributes	11-22
11-18.	Mode 10 Memory Plane Assignments	11-23
11-19.	Mode 10 Base Colors	11-23
11-20.	Palette Register Color Attributes	11-24
11-21.	External Register Values	11-25
11-22.	82C431 Register Values	11-25
11-23.	82C432 Register Values	11-26
11-24.	82C433 Register Values	11-26
11-25.	82C434 Register Values	11-27
12-1.	80386 Register Usage	12-6
12-2.	80386 Internal Registers	12-6
12-3.	80386 Flag Definitions	12-9

Contents

12-4.	Register CR0 Bit Definitions	12-13
12-5.	Interrupt Priorities	12-21
12-6.	80386 Interrupt Vector Assignments	12-21
12-7.	DR6 Register Definitions	12-23
12-8.	Register DR7 Bit Definitions	12-24
12-9.	Test Bit Definitions	12-27
12-10.	Page Directory/Page Table Entries	12-39
12-11.	Page Fault Error Code Bits	12-41
12-12.	Access Rights Byte Bit Definitions	12-44
12-13.	System Segment Type	12-47
12-14.	Device Pinout Grouped by Function	12-60
12-15.	Device Pinout Grouped by Connection	12-61
12-16.	A0, A1 Address Line Generation	12-63
12-17.	Data Duplication as a Function of the Byte Enable Lines	12-63
12-18.	Bus Cycle Definition	12-64
13-1.	80287 Status Word Bit Definitions	13-3
13-2.	80287 Condition Codes	13-4
13-3.	80287 Control Word Bit Definitions	13-6
13-4.	80287 Pin Descriptions	13-11
13-5.	80387 Status Word Bit Definitions	13-15
13-6.	80387 Condition Codes	13-17
13-7.	80387 Quotient Results	13-18
13-8.	80387 Pinout by Function	13-23
13-9.	80387 Pinout by Pin Name	13-24
14-1.	Interrupt Line Assignments	14-2
14-2.	Initialization Command Word 1	14-6
14-3.	Initialization Command Word 2	14-7
14-4.	Initialization Command Word 3	14-7
14-5.	Initialization Command Word 4	14-8
14-6.	Operation Command Word 2	14-19
14-7.	Operation Command Word 3	14-20
14-8.	8259 Interrupt Controller Pin Functions	14-21
14-9.	Timer Control Registers	14-24
14-10.	Timer Read and Write Operations	14-31
14-11.	8254 Programmable Interval Timer Pin Functions	14-32
14-12.	8237 Internal Registers	14-36
14-13.	Command Register	14-38
14-14.	Mode Register	14-39
14-15.	Mask Register	14-40
14-16.	Single Mask Command	14-41
14-17.	Request Register	14-42
14-18.	8237 Pin Functions	14-47
14-19.	Register A Bit Operations	14-54

Contents

14-20.	Periodic Interrupt Time Interval Options	14-54
14-21.	Register B Bit Operations	14-55
14-22.	Register C Bit Operations	14-56
14-23.	Time, Calendar, and Alarm Data Modes	14-57
14-24.	Real-Time Clock Pin Functions	14-62
14-25.	SCP Status Register Bit Definitions	14-65
14-26.	SCP Commands	14-67
14-27.	SCP Command Byte Description	14-69
14-28.	SCP Pin Functions	14-73
15-1.	Memory Configuration	15-1
15-2.	Memory Bank Assignments	15-10
15-3.	Memory Board Error Reporting Scheme	15-14
15-4.	Memory Bank Error Reporting Scheme	15-14
15-5.	EMS I/O Addresses	15-20
16-1.	Floppy Drive Control Register Port Addresses	16-5
16-2.	Digital Output Register	16-6
16-3.	Drive Select	16-7
16-4.	Main Status Register	16-8
16-5.	Floppy Control Register	16-9
16-6.	Floppy Data Registers ST0-ST3	16-10
16-7.	765 FDC Signal Descriptions	16-14
16-8.	Floppy Disk Controller Commands	16-19
16-9.	Controller Command Bits	16-21
16-10.	Read Data Command	16-21
16-11.	Read Deleted Data Command	16-23
16-12.	Write Data Command	16-24
16-13.	Write Deleted Data Command	16-26
16-14.	Read a Track Command	16-27
16-15.	Read Sector ID Command	16-28
16-16.	Format a Track Command	16-29
16-17.	Scan Equal Command	16-30
16-18.	Scan Low or Equal Command	16-31
16-19.	Scan High or Equal Command	16-33
16-20.	Recalibrate Command	16-34
16-21.	Sense Interrupt Status Command	16-34
16-22.	Specify Command	16-35
16-23.	Drive Status Command	16-35
16-24.	Seek Command	16-36
16-25.	Drive Control Register Port Addresses	16-37
16-26.	Diagnostic Mode Error Register Codes	16-38
16-27.	Operation Mode Error Register Codes	16-38
16-28.	Drive and Head Register Codes	16-41
16-29.	Status Register Codes	16-41
16-30.	Command Register Coding	16-43

16-31.	Seek and Restore Step Rate Codes	16-44
16-32.	Fixed Disk Register Codes	16-45
16-33.	Digital Input Register Codes	16-45
16-34.	Format Track Command Track Layout	16-49
16-35.	2 to 1 Sector Interleave	16-51
17-1.	Monochrome Attribute Characteristics	17-12
17-2.	Color Attribute Byte Characteristics	17-12
17-3.	Color Selection	17-13
17-4.	Palette Colors	17-14
17-5.	Mode F Attributes	17-14
17-6.	Mode 10 Memory Plane Assignments	17-15
17-7.	Mode 10 Base Colors	17-16
17-8.	Palette Register Color Attributes	17-16
17-9.	6845 Input/Output Port Address	17-21
17-10.	6845 Data Registers	17-22
17-11.	6845 Register R8, Bits 0 and 1	17-24
17-12.	6845 Register R8, Bits 4 and 5	17-24
17-13.	6845 Register R8 Bits 6 and 7	17-25
17-14.	Video Mode Select Port Register	17-26
17-15.	Video Mode Select Resolution	17-27
17-16.	Color Select Port Register	17-28
17-17.	Palette Definitions	17-28
17-18.	Status Port Register Report	17-29
17-19.	MDA Control Register Functions	17-30
17-20.	Character/Attribute Addressing	17-32
17-21.	Attribute Codes	17-33
17-22.	Character Font	17-35
17-23.	MDA Register Addresses	17-36
17-24.	CRTC Registers and Default Values	17-37
17-25.	CRT Controller Register Functions	17-38
17-26.	CGA I/O Port Assignments	17-43
17-27.	CGA I/O Port Selection	17-44
17-28.	Color Select Register Logic	17-45
17-29.	Palette Number 1 Selection	17-46
17-30.	Palette Number 2 Selection	17-46
17-31.	Mode-Select Port 3D8 Logic	17-47
17-32.	Mode-Select Port 3DA Logic	17-48
17-33.	Status Select Logic	17-49
17-34.	Display Mode Control Register	17-51
17-35.	Display Status Register	17-52
17-36.	Configuration Register	17-54
17-37.	Video Attributes	17-55
17-38.	Graphics Controller Register Summary	17-59
17-39.	Graphics Controller Addressing	17-60
17-40.	Bit to Memory Plane Relationship	17-61

Contents

17-41.	Color Compare Register Example	17-62
17-42.	Graphics Register	17-63
17-43.	Modes of Operation	17-64
17-44.	Shift Register Mode Output Format, Bit 5 is 0	17-65
17-45.	Shift Register Mode Output Format, Bit 5 is 1	17-66
17-46.	Miscellaneous Register	17-66
17-47.	82C431 Graphics Controller Pin Functions	17-68
17-48.	82C432 Sequencer Register Summary	17-73
17-49.	Sequencer Addressing	17-73
17-50.	Clocking Mode Register Functions	17-75
17-51.	Character Map Select A (Bits 0 and 1)	17-77
17-52.	Character Map Select B (Bits 2 and 3)	17-78
17-53.	Memory Mode Register	17-78
17-54.	82C432 Sequencer Pin Functions	17-79
17-55.	82C433 Status Port	17-84
17-56.	82C433 Register Summary	17-84
17-57.	Attributes Controller Addressing	17-85
17-58.	Palette Register and Overscan Register Video	17-87
17-59.	Mode Control Register Functions	17-87
17-60.	Color Plane Enable Register Bits 4 and 5	17-89
17-61.	82C433 Attributes Controller Pin Functions	17-90
17-62.	82C434 CRT Controller Register Summary	17-99
17-63.	CRT Controller Addressing	17-100
17-64.	Overflow Register Contents	17-104
17-65.	Mode Control Register Functions	17-108
17-66.	Mode Register Functions	17-110
17-67.	82C434 CRT Controller Pin Functions	17-110
17-68.	Color Palette Registers	17-119
18-1.	Serial Connector Pinout	18-2
18-2.	NS16450 Register Ports	18-5
18-3.	Baud Rate and Divisor Latch Values	18-6
18-4.	NS16450 Interrupts	18-7
18-5.	Line Control Register	18-8
18-6.	Modem Control Register	18-8
18-7.	Line Status Register Report	18-9
18-8.	Modem Status Register Report	18-10
18-9.	8250 Pin Functions	18-11
18-10.	Parallel Connector	18-17

Listings

6-1.	Sample Code Segment	6-20
7-1.	Function 89 Code Sample	7-11

Part I

Introduction

Chapter 1

Introduction and General Information

The Advanced Desktop Computer, illustrated in Figure 1-1, is a state-of-the-art computer system. It offers greatly increased computing power while maintaining compatibility with earlier PC-compatible computers. Some of the features of this computer are:

- An Intel 80386 microprocessor operating at 16 MHz
- Optional support for either the 80287 or the 80387 coprocessor
- A 101-key keyboard featuring separate numeric and cursor control key groups plus a special emulation mode
- 1M of standard system memory
- Optional memory expansion up to 16M
- Internal hardware support for EMS memory
- EGA, CGA, and MDA video support
- Flexible drive configuration options using standard 5.25-inch or 3.5-inch floppy disk drives and hard disk drives with capacities up to 80M.

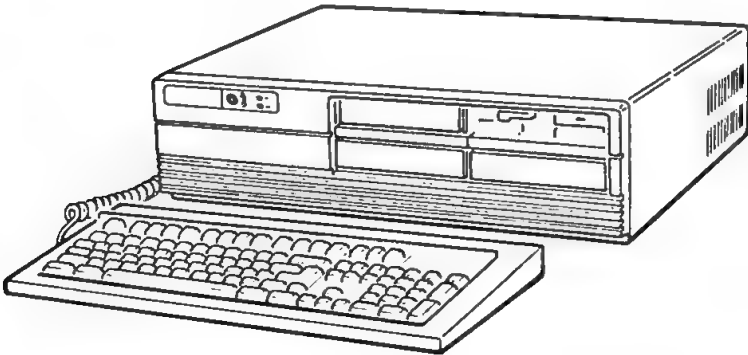


Figure 1-1. Advanced Desktop Computer

Introduction and General Information

Manual Description

This manual contains information on computer operation and programming methods. It is a companion manual for the *MS-DOS Programmer's Utility Pack* (CB-3163-30) and also serves as a hardware introduction for the service technician. Refer to the Owner's Manual for this computer for explanations of commonly used acronyms and terms.

This technical manual contains three parts with appendices and an index:

- Part I contains an introduction to the computer, including information on installation, system configuration, operation, and disassembly.
- Part II describes system programming. This includes the Monitor program and the software-based interrupts common to PC-compatible computers.
- Part III provides information on the base machine configuration and the hardware used to implement that configuration. This includes a discussion of each major programmable integrated circuit in the computer. Part III also includes a chapter on common problems and how to resolve them.
- The appendices contain reference material useful to the system programmer.

Related Publications

Several publications are available which contain information related to the topics discussed in this manual. This material is useful as additional reference material to increase your knowledge and understanding of the computer system.

Introduction and General Information

High-Performance Workstation Owner's Manual (595-3892) — This manual comes with the computer. It describes the installation and operation of the computer from a user's point of view. It also includes a brief discussion of the hardware, configuration, and maintenance.

Intel Introduction to the 80386 — This manual, published by the Intel Corporation, provides an overview and introduction to the 80386 microprocessor. The Intel order number for this publication is 231252.

Intel Microsystems Component Handbook, Vol. 1 & 2 — These manuals, published by Intel Corporation, provide information on the architecture and programming procedures for the 80287 and 80387 numeric processor extensions. They also contain information relating to system support circuitry and microprocessors. The Intel order number for this set is 230843-004.

Intel 80386 Programmer's Reference Manual — This manual, published by Intel Corporation, contains detailed information on the architecture, instruction set, and programming procedures required for the 80386 microprocessor. The Intel order number for this manual is 230985-001.

Intel 80386 Hardware Reference Manual — This manual, published by the Intel Corporation, contains detailed descriptions of the architecture and operation of the 80386 microprocessor. Device timing diagrams and pinouts are also included in this manual. The Intel order number for this publication is 231732-001.

Lotus/Intel Expanded Memory Device Interface Specification — This publication is available directly from Intel Corporation. This specification details the software and hardware interface for EMS memory systems. Intel Technical Support provides a toll free number, 1-800-538-3373, that you may call to obtain this specification.

Intel publications may be obtained by writing to the following address:

Intel Literature Sales
P.O. Box 58130
Santa Clara, CA 95052-8130

Introduction and General Information

Intel also provides a toll-free number for literature requests. The number to call is 1-800-548-4725.

MS-DOS Programmer's Utility Pack (CB-3163-30) — This software package, produced by Microsoft and Zenith Data Systems, contains information and software for writing and assembling MS-DOS and native code 8086, 8088, 80286, and 80386 assembly language programs.

Product Applications

The Advanced Desktop Computer Series provides the user with a great deal of computing power as well as versatility. Because of this, the computer can be useful in a number of different applications.

Professional — The computer uses an Intel 80386 microprocessor and optionally supports the 80287 or 80387 coprocessor. This provides the computing power necessary for professional applications such as CAD/CAM and Desktop Publishing.

Business — The computing power of this machine coupled with a vastly increased memory capacity makes this computer extremely useful in the business community. Spreadsheet applications, databases, and expert systems are a few of the major types of software packages that make this machine important in every day business.

Education — Computing power and large memory support are attractive in other areas as well. This computer is useful in a number of educational applications such as Computer Aided Instruction and simulations.

Personal — This computer also can be useful in personal applications requiring large memory capacities coupled with high computational power. Software development and home business ventures are just two applications available in this area.

Introduction and General Information

Product Information

The following sections detail the operating requirements for this computer as well as the system specifications.

Environment

Most computer designs will operate over a wide range of environmental conditions. Temperature and humidity are the two environmental conditions that most affect the way a computer operates. In general, always be aware of the published specifications for the equipment you are trying to use. Unusual applications may require you to contact the equipment manufacturer for information.

The base system unit is designed to meet working conditions in the average business office (refer to "Specifications" in this chapter). Reasonable control of the temperature and relative humidity while the equipment is operating is essential. Use these guidelines to maintain a proper operating environment for your computer:

- Keep the computer away from sources of heat, including direct sunlight.
- Keep the computer away from moisture sources that might cause liquid to come into contact with the computer. Do not allow drinks or containerized liquids in the work area.
- Install the computer in an area that does not inhibit free air flow around the computer. Restriction of ventilation around the computer can result in overheating problems.
- Computers are extremely sensitive to dust and dirt particles. Consider using protective covers whenever the computer is not in operation. This can help to reduce maintenance calls.
- Computers and floppy disks are also sensitive to electromagnetic radiation. The work area should be free of potential sources of high electromagnetic emissions. Keep floppy disks at least two feet from telephones and associated wiring.

Introduction and General Information

- Computers can be sensitive to mechanical shock and external vibrations. Install the computer on a secure work surface that is free from vibration.

In general, peripherals follow the same guidelines as the computer. Each peripheral you purchase should include specifications and setup instructions. Review these carefully before you operate the equipment. Specific questions should be directed to the equipment manufacturer or representative.

Always place dot-matrix and daisy wheel printers on separate printer stands. These devices are vibration sources and it is best to isolate them from the computer.

Optional Cards

Up to six optional system expansion cards can be installed to enhance the performance of the base system. Two of these cards are described in the following paragraphs.

Memory Expansion — The base system comes with 1M of installed memory. The maximum available memory for the system is 16M. Memory expansion cards are available in increments of 1M and 4M. You may install optional memory using either an extended or EMS memory configuration.

System Cache — An optional system cache card is available to increase the overall throughput of the computer. The amount of performance increase gained by using this card is directly related to the application you are running.

Coprocessors

Coprocessors are integrated circuits that increase system performance in a specific area. Two optional math coprocessors are available for this system: the 80287 and the 80387. Intel refers to these devices as Numeric Processor Extensions. This name, and the term coprocessor refer to the same type of device in this manual. These devices provide vastly increased performance on computation intensive operations. Only one coprocessor may be installed in the system at any given time.

Video Options

The base system provides direct support for EGA video operation. This enhanced level video system provides screen resolution of 640 × 400 pixels with 16 colors from an available palette of 256,000. The system also supports an enhanced EGA mode that supports a higher screen resolution of 640 × 480 pixels. The video system can also support CGA, MDA, and Hercules video display formats.

NOTE: EGA systems require special video monitors to correctly display higher resolution on the screen. Because of the reprogrammed video scan rates, some software packages may not work correctly if run in this enhanced environment. Check with the software manufacturer if you have questions concerning installation or operation.

Mass Storage Options

The term "mass storage" refers to any means of storing large amounts of data for use at a later time. The base system provides the user with one floppy disk drive and a hard disk drive. The following devices are available for this system, any combination of which may appear in the base system:

Introduction and General Information

3.5-Inch Micro Floppy Disk Drives — There are two available drive options in this size format. One drive provides a formatted capacity of 720K bytes of data, the other provides 1.4M of formatted data capacity.

5.25-Inch Floppy Disk Drives — Two different types of drives are available in this size. The standard high-density drive provides a formatted capacity of 1.2M of data per disk. Optional drives are available with either 360K or 1.2M of formatted capacity.

Hard Disk Drives — There are a number of hard disk drives available for this system. Storage capacities for these drives range from 20M to 80M of formatted data. These drives come in either half-height or full-height configurations, depending on the drive vendor.

Removable Cartridge Hard Disk Drives — These drives are similar to the normal hard disk drives with two exceptions. First, the drive is equipped with a removable cartridge similar to a floppy disk. This cartridge contains the platters on which the data is stored. The second difference is in the drive capacity. Each removable cartridge has a maximum storage capacity of 10M of data.

The Advanced Desktop Computer can support a maximum combination of two floppy drives and two hard drives.

Introduction and General Information

Specifications

This section contains design and performance specifications for this computer. Keep these specifications in mind while operating your computer system.

CPU

Processor:	80386, CHMOS technology.
Type:	32-bit internal/external.
Clock speed:	16 MHz.

Memory:	1M dynamic RAM standard, selectable system memory size, 512K or 640K expandable to 16M. Hardware support for EMS memory.
---------	--

Sound:	2 inch speaker.
--------	-----------------

Input/output

Serial port:	Asynchronous serial RS-232C port (AT-style DB-9 connector). One start bit; 7- or 8-bit word length; one or two stop bits; selectable baud rates of 110, 150, 300, 600, 1200, 2400, 4800, or 9600 baud; RD, CTS, DSR, CD signals recognized; TD, RTS, DTR control signals generated; half- or full-duplex operation.
--------------	---

Parallel port:	Centronics-type parallel output port (DB-25 connector).
----------------	---

Video:

MDA, HGC, CGA, and EGA Native Modes

MDA (Monochrome Display Adapter):

Horizontal scan frequency:	18.43 kHz
Vertical refresh rate:	50.0 Hz
Pixel resolution:	720 (H) × 350 (V)
Character cell:	9 × 14

Introduction and General Information

HGC (HERCULES Graphics Card):

Horizontal scan frequency: 18.43 kHz
Vertical refresh rate: 50.0 Hz
Pixel resolution: 720 (H) × 350 (V)
Character cell: 9 × 14

CGA (Color Graphics Adapter):

Horizontal scan frequency: 15.75 kHz
Vertical refresh rate: 60.0 Hz
Pixel resolution: 320 (H) × 200 (V) double scanned
at 400 (V), 2 palettes of 4 colors
each; and 640 (H) × 200 (V),
monochrome.
Character cell: 8 × 16

EGA (Enhanced Graphics Adapter):

Horizontal scan frequency: 21.80 kHz
Vertical refresh rate: 60.0 Hz
Pixel resolution: 640 (H) × 350 (V), 16 colors from
a palette of 64.
Character cell: 8 × 14

31.49 kHz Analog Video Modes:

MDA (Monochrome Display Adapter):

Horizontal scan frequency: 31.49 kHz
Vertical refresh rate: 70.0 Hz
Pixel resolution: 720 (H) × 350 (V)
Character cell: 9 × 14

HGC (HERCULES Graphics Card):

Horizontal scan frequency: 31.49 kHz
Vertical refresh rate: 70.0 Hz
Pixel resolution: 720 (H) × 350 (V)
Character cell: 9 × 14

Introduction and General Information

CGA (Color Graphics Adapter):

Horizontal scan frequency: 31.49 kHz
Vertical refresh rate: 70.0 Hz
Pixel resolution: 320 (H) × 200 (V) double scanned at 400 (V), 2 palettes of 4 colors each; and 640 (H) × 200 (V), double scanned to 400 (V), monochrome.
Character cell: 8 × 16

EGA (Enhanced Graphics Adapter):

Horizontal scan frequency: 31.49 kHz
Vertical refresh rate: 70.0 Hz
Pixel resolution: 640 (H) × 350 (V), 16 colors from a palette of 256,000.
Character cell: 8 × 14

Zenith Mode (Zenith 480 Line):

Horizontal scan frequency: 31.49 kHz
Vertical refresh rate: 60.0 Hz
Pixel resolution: 640 (H) × 480 (V), 16 colors from a palette of 256,000.
Character cell: 8 × 19

Video Memory Capacity 256K

Monitor Interface: 15-pin connector: RGB analog video, constant 31.49 kHz scan frequency, 75 ohm impedance, 0-0.714 volt full scale, zero volts represents black.

9-pin connector: TTL video with native horizontal scan frequencies in RGBI, RGBrgb, V (Video), or VI (Video, Intensity) modes.

Introduction and General Information

Disk drives:

5.25-inch double-sided, double-density floppy disk drives, 360K formatted capacity each drive, 48 tpi, nine sectors per track. 5.25-inch double-sided, quad-density floppy disk drives, 1.2M formatted capacity each drive, 96 tpi, nine sectors per track.

3.5-inch double-sided, double-density micro floppy disk drives, 720K formatted capacity per drive, 80 tpi, nine sectors per track. 3.5-inch double-sided, quad-density micro floppy disk drives, 1.44M formatted capacity per drive, 80 tpi, 18 sectors per track. Write-protection recognized. Optional fixed or removable cartridge hard disk drives. Capacity: 10M to 80M. Operating system: MS-DOS version 3.

Keyboard:

101 keys: 58-key alphanumeric typewriter arrangement (duplicate CTRL, ALT, and SHIFT keys), 17-key numeric keypad, 4-key cursor control pad, 6-key editing pad, 3-key function pad, and 12 programmable function keys. Full 84-key PC keyboard compatibility maintained by using a switchable keyboard emulation mode. CAPS LOCK, SCROLL LOCK, and NUM LOCK keys use status LEDs.

Introduction and General Information

Power requirements:	120 VAC at 4 A (200 W) or 240 VAC at 2 A (200 W).
Input frequency:	120-V: 50 or 60 Hz (48 – 62 Hz). 240-V: 50 or 60 Hz (48 – 62 Hz).
Environment	
Operating:	60° – 85° F (16° – 30° C) at 20% – 80% relative humidity (non-condensing).
Storage:	–40° – +125° F (–40° – +51°C) at 20% – 80% relative humidity (non-condensing).
Dimensions:	21.0 in. wide × 16.25 in. deep (front to back) × 6.25 in. tall (53.34 cm × 42.54 cm × 15.88 cm).
Weight:	41 lbs (18.85 kg) base system configuration.



Chapter 2

Installation

This chapter discusses connecting the computer to peripherals and power in preparation for use and installing internal options.

Required Tools

If you are connecting the computer to peripherals and power, you only need a slotted screwdriver. If you are installing internal options, you will need the following tools:

- Slotted screwdriver
- 1/4-inch nut driver
- No.2 phillips screwdriver.

Operating Environment and Power Requirements

- The room temperature should be between 60° and 85° F (16° to 30° C).
- The relative humidity should be between 20% and 80% (non-condensing).
- Allow six inches of unobstructed space at the back panel and on the ventilation slot side of the computer to prevent overheating. The computer may be operated on its side, fan exhaust side up, if you allow the required ventilation space. Use a permanent mounting for stability.
- The computer is factory-set for 115 VAC. Change the line voltage select switch, located on the back panel, if you are in an area serviced by a 230 VAC power source and contact your service representative to obtain the proper power cord. The cord packed with your computer is intended for use with 115 VAC only.

Installation

If you use a switched power strip or multiple-outlet box for all your components, be sure it meets the following power requirements:

- For 115 VAC systems, a minimum power rating of 10 amperes is required.
- For 230 VAC systems, a minimum power rating of 5 amperes is required.
- Do not use an extension cord unless it is a heavy-duty, three-wire type. Smaller cords can pose a safety hazard and tend to reduce the amount of voltage available, causing performance problems with your computer.

Unpacking and Setting Up

1. Carefully unpack your computer and place it on the work surface. Your operating system documentation and disks are packed on top of the computer. The keyboard and power cord are packed under the computer.
2. Position your computer so you have access to the back panel to check the switches and connect accessories. Allow six inches of unobstructed open space around the ventilation slots. Figure 2-1 shows the back panel of your computer and the ventilation slots.

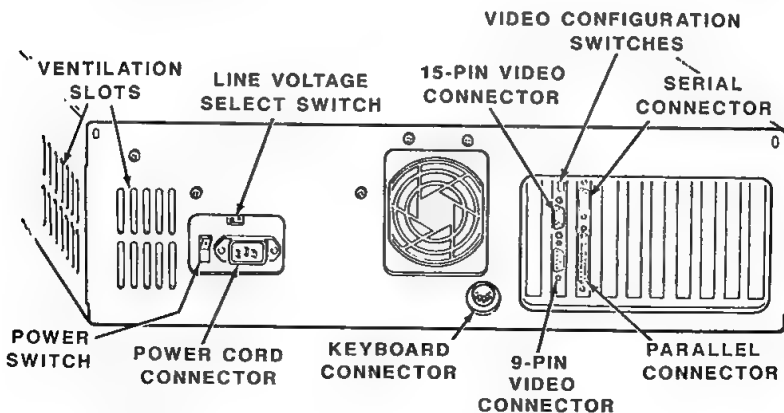


Figure 2-1. The Back Panel of the Computer

3. Check the switches on the back panel to make sure that:
 - The power switch is in the OFF position. The white dot indicates the ON position.
 - The line voltage select switch setting matches your AC power source.
4. If you want to install internal options, refer to "Installing Internal Options" in this chapter and install them now.

Connecting Peripherals

1. Your computer is factory-set to operate with either a high-resolution analog (31 kHz) video monitor or a multisync video monitor. If you want to use a standard EGA, color, or monochrome video monitor, refer to Table 2-1 and reconfigure your video controller card now. Refer to Chapter 3 for disassembly instructions and Chapter 4 for detailed configuration instructions.

Table 2-1. Video Controller Card Configuration

MONITOR TYPE	VIDEO DIP SWITCH SECTIONS						JUMPERS	
	1	2	3	4	5	6	P1	P3
Monochrome ¹	OFF	OFF	ON	OFF	ON	OFF	2&3	1&2
Color (80 × 25)	OFF	OFF	OFF	ON	ON	OFF	2&3	1&2
Color (40 × 25)	ON	OFF	OFF	ON	ON	OFF	2&3	1&2
Enhanced RGB (200 line)	ON	ON	ON	OFF	ON	OFF	1&2	1&2
Enhanced RGB (350 line)	OFF	ON	ON	OFF	ON	OFF	1&2	1&2
High-resolution analog monochrome	ON	ON	OFF	OFF	ON	ON	1&2	1&2

Installation

Table 2-1 (continued). Video Controller Card Configuration

MONITOR TYPE	VIDEO DIP SWITCH SECTIONS						JUMPERS	
	1	2	3	4	5	6	P1	P3
High-resolution analog color or multi-sync EGA ²	OFF	ON	ON	OFF	ON	ON	1&2	1&2

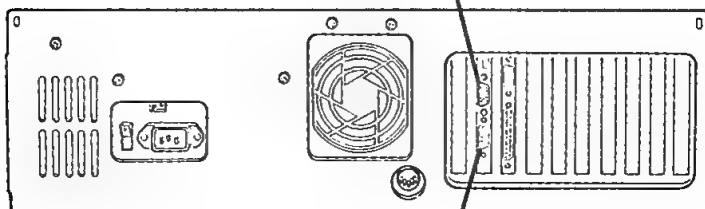
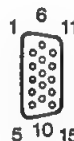
NOTES:

1. Some monochrome monitors operate with color signals rather than monochrome signals. Check your monitor documentation. If the monitor uses RGB signals, set the DIP switch for the appropriate color or enhanced color mode.
2. Factory setting.

Sections 1 through 4 of the video controller card DIP switch determine the video signals on the 9-pin video connector. Section 5 enables automatic emulation of CGA, HGC, MDA, and EGA video modes, as required by software, when it is in the ON position. Switch section 6 controls the video output signals. When this switch is on, analog video signals at the 31 kHz horizontal sweep frequency are available at the 15-pin connector. If the switch is off, TTL-level video signals are available at the 9-pin connector. Figure 2-2 provides pin definitions for the 15-pin and 9-pin video connectors.

15-PIN CONNECTOR FOR 31 KHZ ANALOG MONITORS

Pin No.	Signal	Pin No.	Signal	Pin No.	Signal
1	Red output	6	Red ground	11	NC
2	Green output	7	Green ground	12	NC
3	Blue output	8	Blue ground	13	Horiz. sync
4	NC	9	NC	14	Vert. sync
5	Ground	10	Sync ground	15	NC



9-PIN CONNECTOR FOR RGB-TTL MONITORS

MONOCHROME

RGB

ENHANCED
RGB

Pin No.	Signal	Pin No.	Signal	Pin No.	Signal
1	Ground	1	Ground	1	Ground
2	Ground	2	Ground	2	S. red
3	Red	3	Red	3	Red
4	Green	4	Green	4	Green
5	Blue	5	Blue	5	Blue
6	Intensity	6	Intensity	6	S. green
7	Video	7	NC	7	S. blue
8	Horiz. sync	8	Horiz. sync	8	Horiz. sync
9	Vert. sync	9	Vert. sync	9	Vert. sync



Figure 2-2. The Video Connectors

Installation

2. Refer to Figure 2-3 and connect the video cable to the appropriate video connector on the back panel of the computer. Use the 15-pin video connector for a high-resolution analog video monitor. Use the 9-pin connector for a multisync, EGA, CGA, or monochrome video monitor. Secure the connection by tightening the screws on the connector.

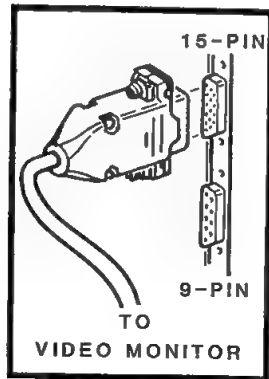


Figure 2-3. Connecting the Video Monitor

3. Connect the power cord to the IEC 3-prong connector and the keyboard cable to the 5-pin DIN-type connector. Both are located on the back panel. Figure 2-4 provides pinout information for the keyboard connector.

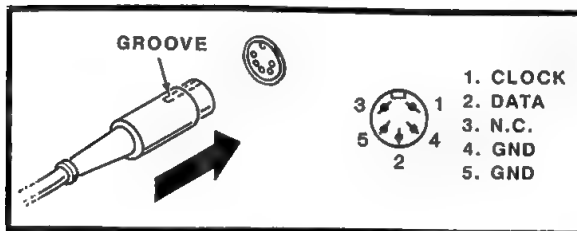
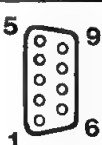
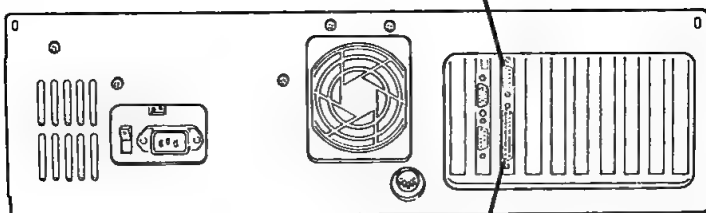


Figure 2-4. The Keyboard Connector

Installation

4. If you want to connect any peripherals to your computer, connect them now. Your computer is equipped with a serial port (COM1) and a parallel port (LPT1). Both are accessible through the back panel at slot 3. The serial port connector is a 9-pin D-type connector, configured as an RS-232C communications port. The parallel port connector is a 25-pin D-type connector, configured as a Centronics-type printer port. Figure 2-5 provides pinout information for serial and parallel ports on the computer.

Pin No.	Signal	Pin No.	Signal
1	Carrier Detect	6	Data set ready
2	Receive data	7	Request to send
3	Transmit data	8	Clear to send
4	Data terminal ready	9	Ring indicate
5	Ground		

Pin No.	Signal	Pin No.	Signal
1	Strobe	10	Acknowledge
2	Data bit 0	11	Busy
3	Data bit 1	12	Page end
4	Data bit 2	13	Select
5	Data bit 3	14	Auto feed
6	Data bit 4	15	Error
7	Data bit 5	16	Initialize printer
8	Data bit 6	17	Select input
9	Data bit 7	18-25	Ground

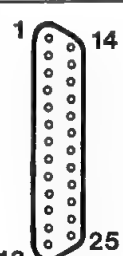


Figure 2-5. The Serial/Parallel Connectors

Installation

5. Position the computer so you can easily reach the disk drives to insert and remove floppy disks. You should also be able to reach the power switch on the back panel. Plug the computer and video monitor power cords into an AC power source. Be sure the power switches are in the OFF position and plug any peripherals into an AC power source.
6. Open the drive doors on the floppy disk drives, and then remove the cardboard shipping inserts. Save the inserts. If you ship the computer in the future, insert them in the disk drive slot to prevent damage to the drive.

Initial Powerup

If your computer system is connected to a switched power strip or multiple-outlet box, a single switch controls power to the entire system. For individual operation, turn on power to the video monitor first, then the computer. Turn on power to any peripherals last. This will allow the peripherals to receive the correct configuration information from the system. As the system begins the powerup sequence it will initiate a series of self-tests.

Self-Testing

The automatic internal self-test sequence checks the major circuits and verifies that various system functions operate properly. If some part of the computer fails to operate correctly, the computer attempts to display an error message on your video monitor. Chapter 19 contains detailed information on the self-tests and error messages.

Autobooting

Once the self-tests have been concluded, the computer will autoboot (automatically load the operating system) from the first hard disk drive (drive C) if your system is configured for this option. Since the operating system has not been installed on the hard disk yet, the following error message will appear on the screen:

Not a bootable partition

When this message appears, enter the Monitor program, insert your MS-DOS SETUP disk in the floppy drive, and boot manually.

NOTE: If no hard disk is present in the system, and the autoboot feature is enabled, the computer will attempt to boot from a floppy disk in drive A.

Entering the Monitor Program

To enter the Monitor program, press the CTRL (control), ALT (alternate) and INS (insert) keys in sequence, hold them down briefly, and then release them. A message similar to the following will be displayed on your video monitor:

```
MFM-300 Monitor, Version x.xx
Memory Size: xxxK + xxxxK + 64K Cache
Enter "?" for help.
->
```

Your computer will display version and memory size numbers in place of the x's shown. If for some reason the Monitor prompt does not appear on your screen, press the CTRL, ALT, and ENTER keys in sequence, hold them down briefly, and then release them. After entering the Monitor program using this key combination, the CPU register contents and the Monitor prompt (->) will be displayed on the screen in a message similar to the following:

```
AX=0EB8 BX=0000 CX=0001 DX=007F SI=0036 DI=035E BP=04C6 SP=04C6
CS=0070 DS=0040 SS=0EB8 ES=0EB8 IP=0E62 FL= NC PE NA ZR PL DI UP NV
0070: 0E62 8BEC MOV BP, SP
```

Refer to Chapter 6 for more information on the Monitor program.

Installation

Booting Manually

The boot command in the Monitor program can load the operating system from either of the two floppy disk drives or any one of four hard disk drives (W0 through W3). (The base system configuration only permits the installation of a maximum of two hard disk drives.) Any one of four hard disk partitions (1 through 4) may also be specified with this command. The syntax for the boot command is:

B<drive type><drive number><:partition number>

If you attempt to boot from a floppy drive that does not contain a system disk, this error message will appear on the screen:

+++ DISK ERROR: Drive not ready! +++

If you attempt to boot from a drive that is not installed in your computer, one of these error message will appear on the screen:

+++ DISK ERROR: Bad disk controller! +++

+++ DISK ERROR: Drive not ready! +++

Loading the Operating System

The operating system links your application programs or any other software with your computer to perform tasks. Either DOS-based (such as MS-DOS) or UNIX-based (such as XENIX) operating systems may be used with this computer. Consult your service representative for additional information on the operating system best suited to your needs.

Not all operating systems load into the computer in the same way. Always follow the start-up instructions provided in your operating system manual. In the case of the MS-DOS operating system, follow this procedure:

1. Locate the operating system documentation and floppy disk(s) shipped with your computer. Insert the operating system SETUP program disk into drive A. Consult your operating system documentation if you are unsure which disk contains the SETUP program.

2. Use the instructions in the "Bootting Manually" section of this chapter and manually boot the system from drive A. Drive A is identified as F0 by the Monitor program. When the opening message appears on the screen, the bootting process is complete.
3. Refer to your operating system documentation as needed and follow the instructions provided by the SETUP program. This program will help you make back-up copies of the floppy disks and install the operating system on your hard disk drive.

Resetting the Computer

To verify that the operating system's SETUP program was executed properly, take the computer through the autoboot process. Press the CTRL, ALT, and DEL keys in sequence, hold them down, and then release them. This key combination resets your computer and causes it to autoboot if the autoboot configuration option is enabled.

Installing Internal Options

Several options are available for your computer to enhance its performance and customize it for your particular application. These options include:

- A Z-416-2 (80287) or a Z-516 (80387) numeric coprocessor.
- A Z-525 cache memory card.
- A Z-505 (1 Megabyte) or a Z-515 (4 Megabyte) memory expansion card.
- Several drive expansion options, including a 360K floppy disk drive, a 40M hard disk drive and an 80M hard disk drive. Consult your dealer for the drive expansion option most suited to your needs.

Installation

Most options that you install in your computer are accompanied by their own documentation. After using the following installation instructions, refer to the option's documentation for configuration and operation information. Before any options can be installed in your computer, you must partially disassemble the computer. Refer to Chapter 3 for disassembly instructions.

WARNING

Potentially lethal voltages are present inside your computer whenever the power cord is connected to an AC source. Disconnect the power cord and all peripheral devices from your computer before attempting disassembly.

CAUTION

Hard disk drives are very fragile and can easily be damaged. Run the MS-DOS SHIP utility (or other similar operating system utility that parks the read/write heads of your hard disk drive in the shipping zone) before disassembling your computers.

Installing a Circuit Card

Figure 2-6 shows the locations of the standard and optional circuit cards, power supply, and disk drive chassis. To operate correctly, the standard circuit cards must remain in their assigned slots.

1. If your computer has a hard disk drive installed, run the SHIP utility .
2. Remove the cover. Refer to Chapter 3 if you are unfamiliar with disassembly procedures.
3. Use the information in Table 2-2 to select the appropriate slot for your new card. Remove and save the screw that secures the blank panel or the card in that slot.

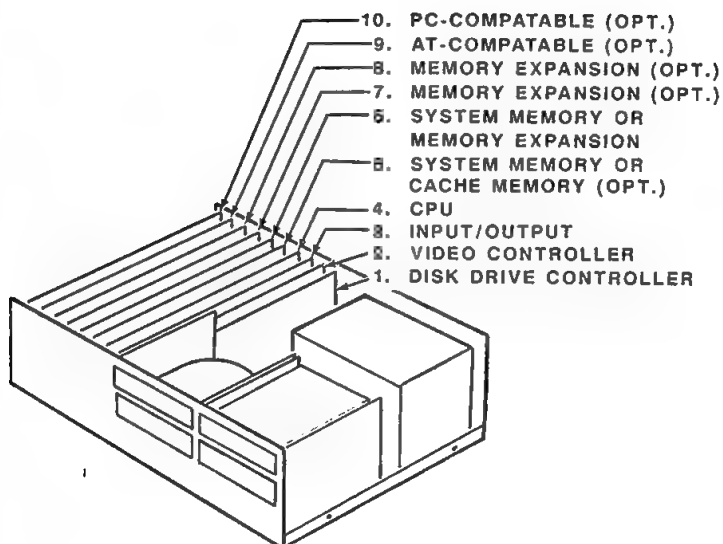


Figure 2-6. Standard and Optional Card Locations

Table 2-2. Card Slot Assignments

SLOT NUMBER	CARD
1	Disk drive controller card
2	Video controller card
3	Input/output card
4	CPU card
5	System memory or optional cache memory card (Z-525)
6	System memory card or optional expansion memory
7	Optional expansion memory card
8	Optional expansion memory card
9	Optional AT-compatible card
10	Optional PC-compatible card

4. Lift the blank panel or card up and out of the computer.
5. Refer to Chapter 4 and configure the jumper blocks or switches, as necessary.

Installation

6. Position the edge connector of the new card directly over the slot on the backplane board. Apply gentle downward pressure until the card is firmly seated in the slot on the backplane board.
7. Secure the card to the chassis using the screw you removed earlier.
8. Reconnect any internal cables. Make sure they do not extend above the top of the disk drive chassis.
9. Replace the cover and secure it with the six screws that you removed and set aside with the cover when you began disassembling your computer.
10. Reconnect your power cord, video monitor, keyboard, and any peripherals to your computer and then to their AC power source.

Installing an Integrated Circuit

The coprocessor and expansion ROM options require the installation of an integrated circuit (IC) onto the CPU card. Use the following precautions when handling ICs to prevent damaging them:

- Equalize the static electricity between the work surface and IC.
 - Do not lay an IC down or let go of it until it is installed on the circuit card or placed in protective packaging.
 - Do not attempt to install an IC unless the pins are aligned. Incorrect installation may result in intermittent contact and can damage the IC pins or the socket.
1. If your computer has a hard disk drive installed, run the SHIP utility.
 2. Remove the cover using the procedure in Chapter 3.

- Remove the CPU card from slot 4. Use the information in Table 2-3 to select the appropriate socket and install the optional IC. Figure 2-7 illustrates the location of the optional IC sockets and proper installation procedures.

Table 2-3. Optional IC Locations

OPTIONAL IC	SOCKET NUMBER
Expansion ROM (64K)	U255
Z-416-2 (80287 coprocessor)	U256
Z-516 (80387 coprocessor)	U213

- Refer to Chapter 4 for configuration information concerning these devices.

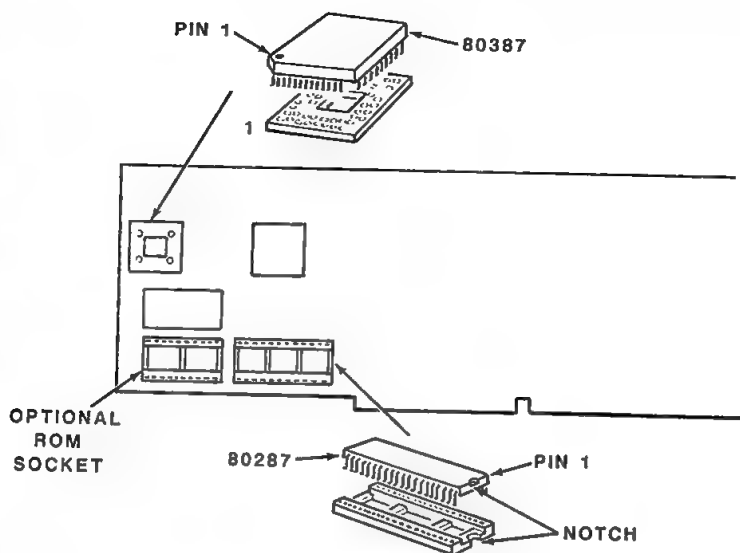


Figure 2-7. Installing Optional ICs



Chapter 3

Disassembly

The following procedures describe how to disassemble the computer. Use these procedures whenever you add optional equipment to the computer system.

CAUTION

If a hard disk drive is installed in the computer, run the MS-DOS SHIP utility prior to disassembly. Failure to do so can result in damage to the drive. If you are not using MS-DOS, run the equivalent operating system utility before disassembly.

Cover Removal

WARNING

Hazardous voltages are present inside the computer whenever it is connected to an AC power source. Disconnect the power cord from the AC power source before disassembling the computer.

Refer to Figure 3-1 and perform the following procedure:

1. If a hard disk drive is installed in the system, turn on the computer and run the MS-DOS SHIP utility. Otherwise, disconnect the power cord and proceed to step 3.
2. Turn off the computer and disconnect the power cord from the AC power source.
3. Disconnect all other devices attached to the computer.
4. Turn the keyboard-disable lock to the fully clockwise position (unlocked).

Disassembly

5. Remove the six screws that secure the cover to the unit.
6. Remove the cover by sliding it toward the front of the computer.

NOTE: Support the cover to avoid contact with circuit cards, cables, or connectors.

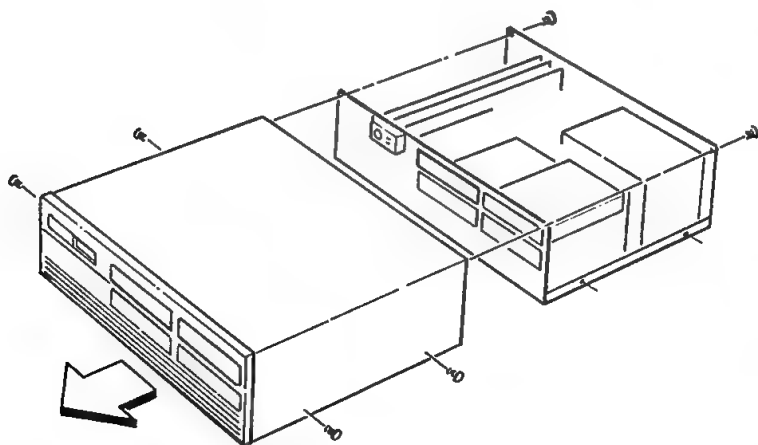


Figure 3-1. Cover Removal

Circuit Card Removal

CAUTION

Turn off the computer and disconnect the power cord from the AC power source.

Refer to Figure 3-2 and perform the following procedure:

1. Remove the computer cover as previously described.
2. Disconnect all internal and external cables from the card to be removed.
3. Remove and retain the support bracket screw, as shown in Figure 3-2.

4. Disengage the card from its edge connector and remove it from the computer.

NOTE: Electronic components on some cards are sensitive to transient static electricity. Observe standard precautions for all circuit cards.

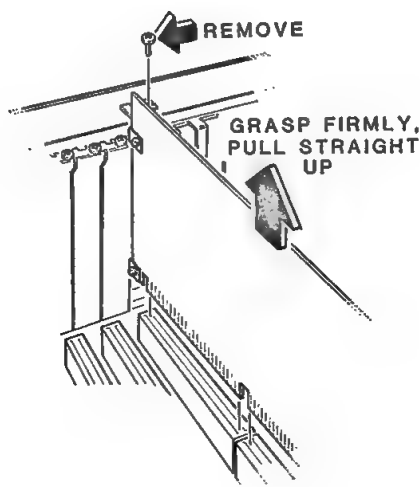


Figure 3-2. Circuit Card Removal

Auxiliary Fan Removal

CAUTION

Turn off the computer and disconnect the power cord from the AC power source.

Refer to Figure 3-3 and perform the following procedure:

1. Remove the computer cover as previously described.
2. Disconnect the fan from its electrical connector on the back-plane board.

Disassembly

3. Remove the two screws that secure the fan assembly to the back panel of the computer chassis.
4. Lift the fan assembly out of the computer.

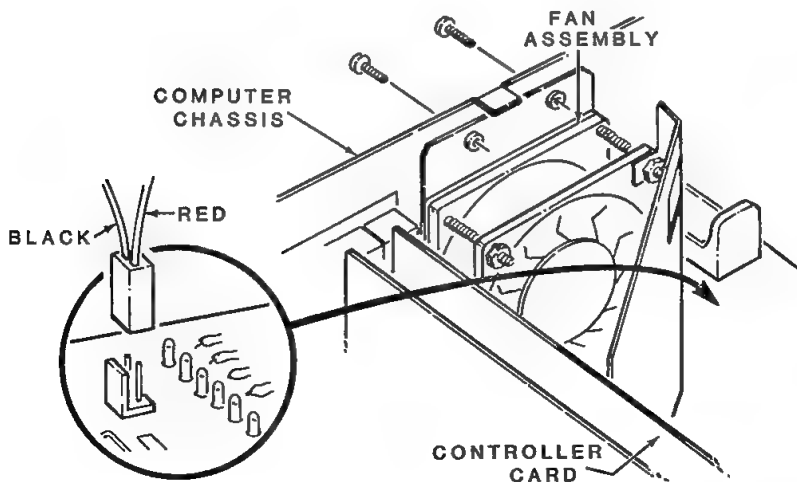


Figure 3-3. Auxiliary Fan Removal

Backplane Board Removal

CAUTION

Turn off the computer and disconnect the power cord from the AC power source.

Refer to Figures 3-4 through 3-7 and perform the following procedure:

1. Remove the computer cover as previously described.
2. Remove all circuit cards as previously described.
3. Remove the auxiliary fan assembly as previously described.

4. Disconnect power supply connector P1 from backplane connector P128, as shown in Figure 3-4.

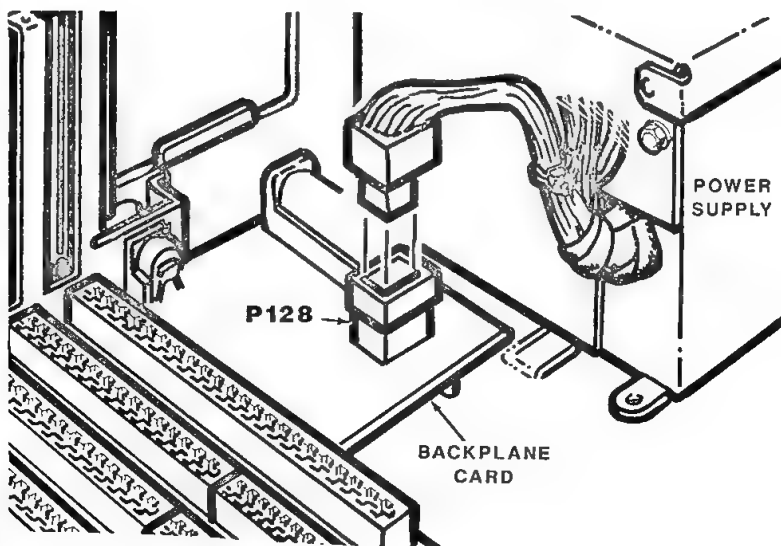


Figure 3-4. Power Supply Connection

Disassembly

5. Disconnect the power supply ground lead from the computer chassis, as shown in Figure 3-5.

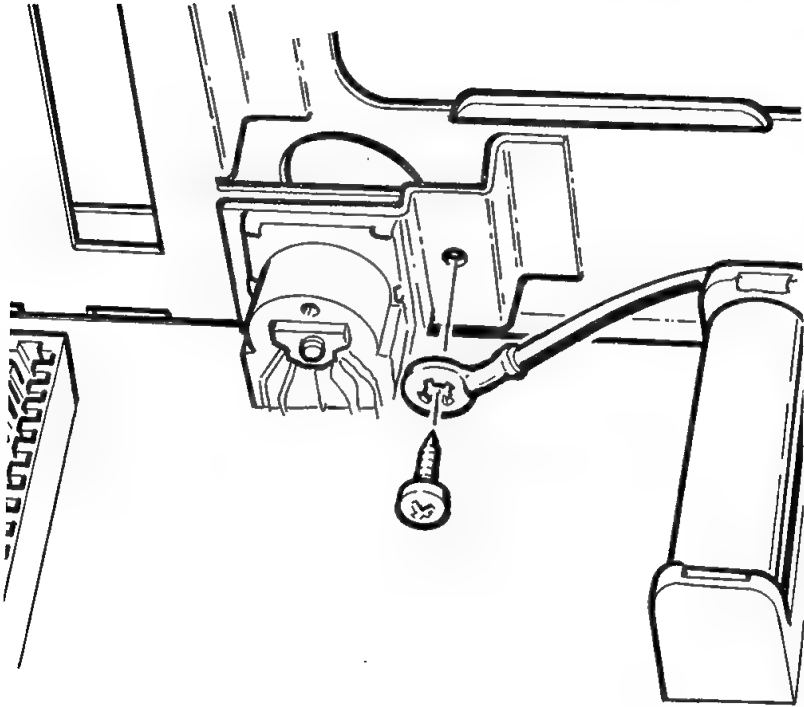


Figure 3-5. Power Supply Ground Lead

6. Remove the ten screws and lift the backplane board out of the computer.

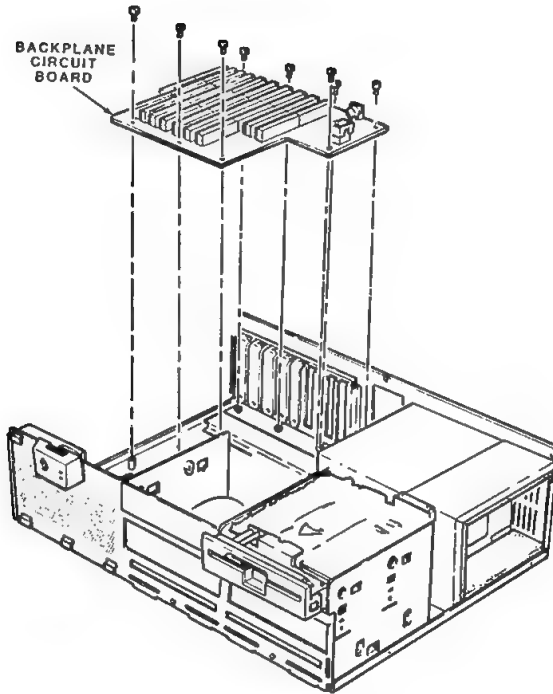


Figure 3-6. Backplane Board Removal

Disassembly

NOTE: Observe the placement of the chassis spring and chassis spring retainer. Figure 3-7 illustrates the spring and retainer. These parts must be replaced when the backplane board is installed.

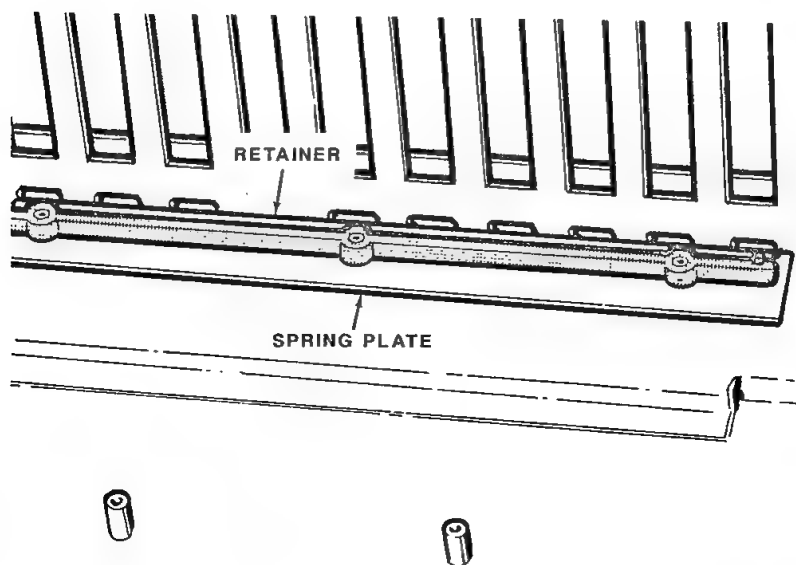


Figure 3-7. Spring and Retainer

Power Supply Removal

CAUTION

Turn off the computer and disconnect the power cord from the AC power source.

Refer to Figures 3-4, 3-5, 3-8, and 3-9 to perform the following procedure:

1. Remove the computer cover as previously described.
2. Remove the auxiliary fan assembly as previously described.
3. Disconnect power supply connector P1 from backplane connector P128, as shown in Figure 3-4.
4. Disconnect the power supply ground lead from the computer chassis, as shown in Figure 3-5.
5. Disconnect power cables from the disk drives, as shown in Figure 3-8.

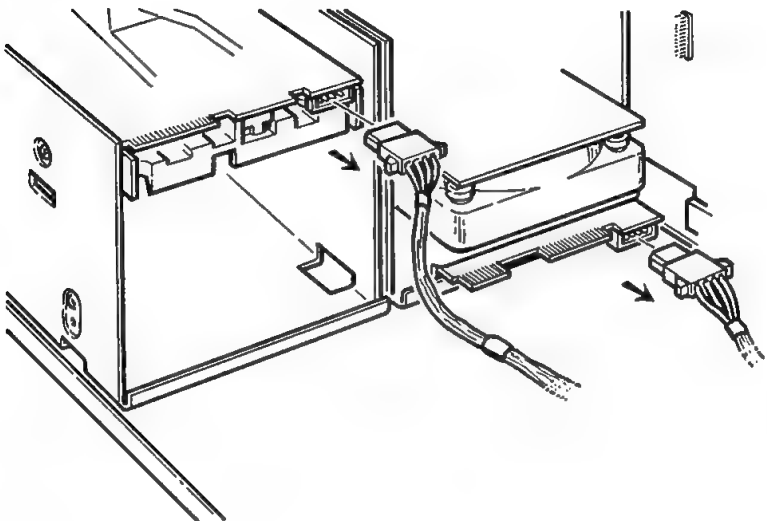


Figure 3-8. Drive Power Connections

Disassembly

6. Remove the three screws that secure the power supply to the back panel of the computer chassis.
7. Remove the screw (beneath the auxiliary fan assembly) that secures the power supply to the bottom of the computer chassis.
8. Slide the power supply about one-half inch toward the front of the computer.

NOTE: If one or more hard disk drives are installed in the computer, it may be necessary to remove them to allow enough clearance for the power supply. Refer to the next section.

9. Lift the power supply up and out of the computer chassis, as shown in Figure 3-9.

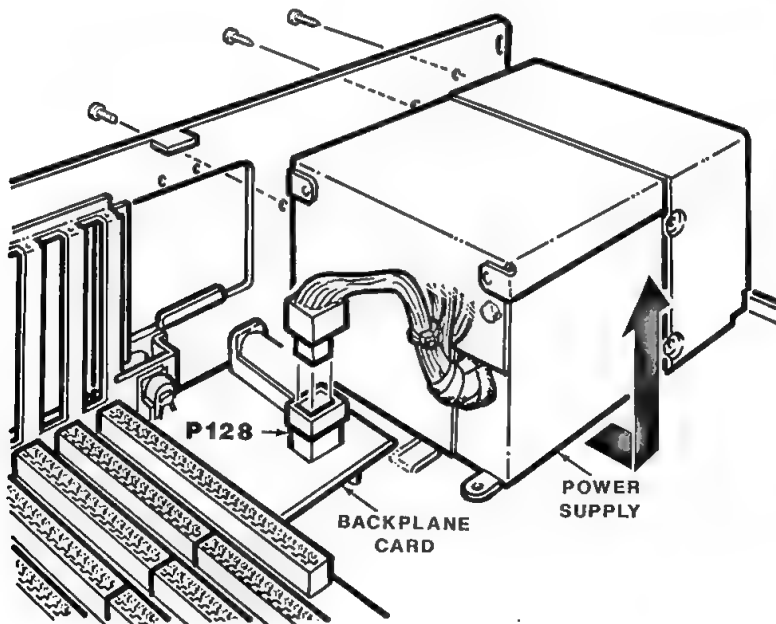


Figure 3-9. Power Supply Removal

Disk Drive Removal

CAUTION

If a hard disk drive is installed in the computer, run the MS-DOS SHIP utility prior to disassembly. Failure to do so can result in damage to the drive. If you are not using MS-DOS, run the equivalent operating system utility before disassembly.

Up to two floppy disk drives and two hard disk drives may be present in the computer. One floppy and one hard disk drive can be mounted in each drive support bracket. Removal of the support bracket assembly allows access to each drive.

Refer to Figure 3-10 and perform the following procedure:

1. If a hard disk drive is installed, run the MS-DOS SHIP utility.
2. Turn off the computer and disconnect the power cord from the AC power source.
3. Remove the computer cover as previously described.
4. Disconnect the power and control signal connectors from each drive to be removed.
5. Remove the support bracket assembly screw from the front of the computer chassis.
6. Slide the support bracket assembly about one-half inch toward the back of the computer.

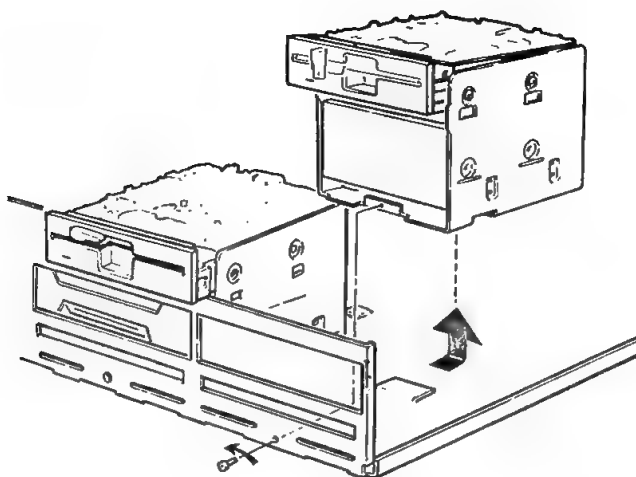


Figure 3-10. Disk Drive Removal

7. Lift the support bracket assembly up and out of the computer chassis.
8. Remove four screws from the appropriate disk drive in the support bracket assembly.
9. Carefully separate the drive from the support bracket assembly.

Chapter 4

Hardware Configuration

Various types of switches and jumpers provide user-selected configuration options on the system circuit cards. In addition, MS-DOS and the system firmware provide configurable options. This chapter describes only the hardware configuration selections available to the user.

CPU Card

The CPU card, shown in Figure 4-1, contains two 8-section switches and five jumpers. Each of these components has a specific function for each switch position and jumper setting.

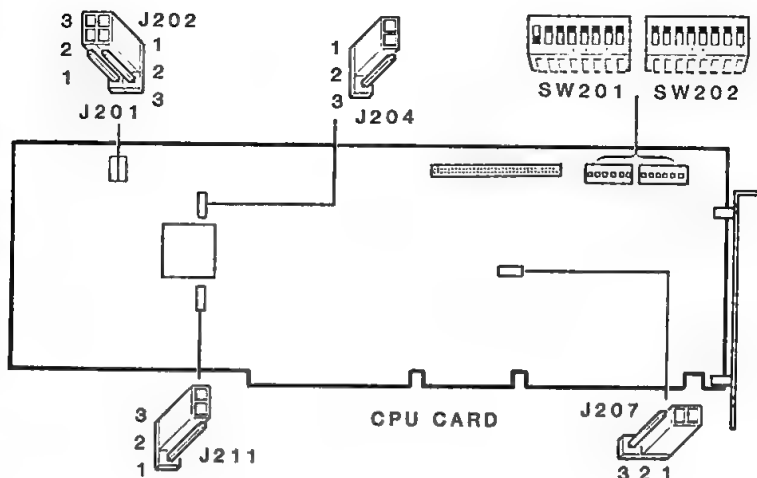


Figure 4-1. CPU Switch and Jumper Positions

Switches SW201 and SW202, shown in Figure 4-1, determine the memory configuration enabled for the computer. The memory configuration of the computer requires that the first memory board be

Hardware Configuration

a Zenith board. Whenever you add a new 32-bit, fast, page-mode accessed memory card to the system, reconfigure the corresponding switches according to the information in Figure 4-2.

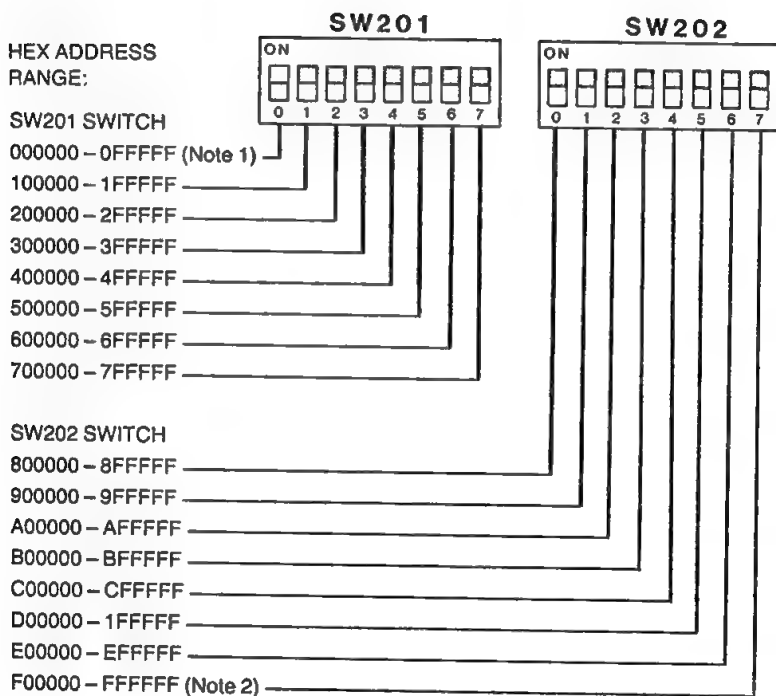


Figure 4-2. Memory Card Base Address Selection

Hardware Configuration

-
- NOTES:**
1. Memory present in the range F00000 – FFFFFFFF must be Zenith 32-bit memory.
 2. You may use third-party memory cards provided you install them on 1M boundaries that do not conflict with Zenith memory. (Third-party cards may NOT occupy the last 1M block.)
 3. It is not necessary to assign memory to continuous 1M blocks.
 4. A Zenith memory card must occupy the first 1M block of memory. The 512K – 640K segment may be "mapped out" to make this area available to third-party cards that use these addresses.
 5. When a switch is in the OFF position this enables the Zenith memory board located in that range. The ON position indicates that no memory exists in that range, or that a third party memory card is in use in that range.
-

Table 4-1 describes the various jumper options on the CPU card. Figure 4-1 illustrates the factory settings.

Table 4-1. CPU Card Jumpers

JUMPER NUMBER	CONFIGURATION OPTION	
	PINS 1-2	PINS 2-3
J201	80387 Synchronous	80387 Asynchronous*
J202	80287 Installed*	80387 Installed
J204	No Coprocessor*	Coprocessor Installed
J207	1M RAM Board Used*	Only 4M RAM Boards Used
J211	Low Speed Coprocessor	High Speed Coprocessor*

*Factory setting

J201 — Allows the bus interface unit in the 80387 coprocessor to operate in asynchronous mode with the CPU clock. This is desirable when the clock input to the CPU is higher than the manufacturer's recommended clock input to the coprocessor.

Hardware Configuration

J202 — Disables the system bus data buffers during the coprocessor cycle if an 80387 is present. If the 80287 is present, it will enable the buffers. This is necessary since the 80387 connects directly to the CPU, while the 80287 connects to the system bus via the buffers.

J204 — Informs the coprocessor interface controller if a coprocessor is available so the necessary handshaking operations can take place.

J207 — Selects the page size for Zenith fast, 32-bit, dynamic memory.

J211 — Connects either a 32 MHz or 16 MHz clock to the coprocessor. For the 80287, the system internally divides the clock frequency by three, so the actual frequency is either 10.67 or 5.33 MHz. For the 80387, the system internally divides the clock frequency by two, so the actual frequency is 16 or 8 MHz.

I/O Card

The system I/O card, illustrated in Figure 4-3, contains all circuitry involved with input and output operations. There are six jumpers on this card that the user may configure. Table 4-2 describes the functions of these jumpers.

Hardware Configuration

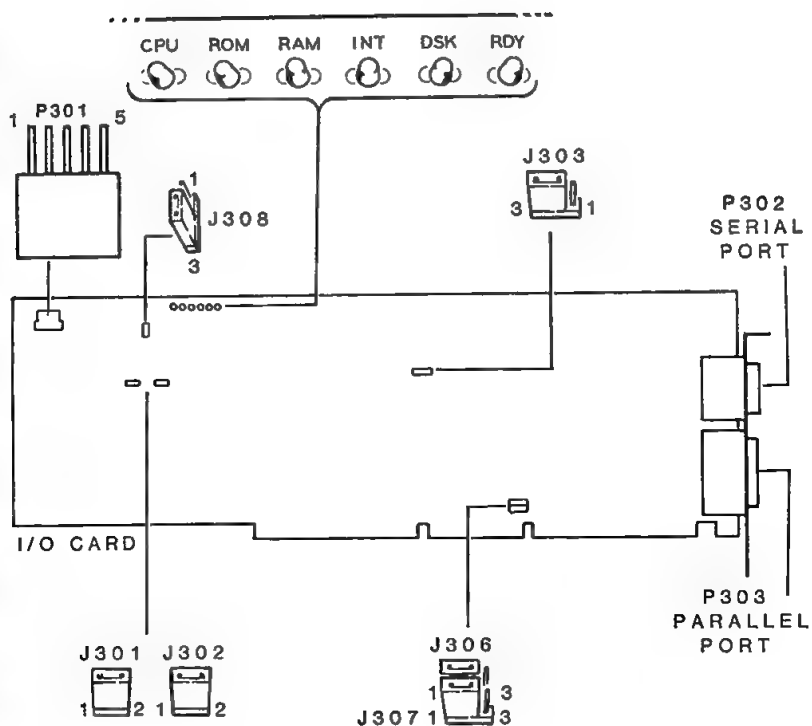


Figure 4-3. I/O Card Configuration Jumpers

Hardware Configuration

Table 4-2. I/O Card Jumpers

JUMPER	FUNCTION
J301	Monochrome/color adapter select Jumper installed: color ¹ Jumper not installed: monochrome
J302	ACOK enable Jumper installed: ACOK generates a non-maskable interrupt, if AC power fails. ¹ Jumper not installed: ACOK ^{1,2}
J303	Serial/parallel port address select ^{2,3} Jumper on pins 1 and 2: port LPT2, COM2 selected Jumper on pins 2 and 3: port LPT1, COM1 selected ¹
J306	Parallel port interrupt select ^{2,3} Jumper on pins 1 and 2: LPT2 interrupt selected Jumper on pins 2 and 3: LPT1 interrupt selected ¹
J307	Serial port interrupt select ^{2,3} Jumper on pins 1 and 2: COM2 interrupt selected Jumper on pins 2 and 3: COM1 interrupt selected ¹
J308	Refresh burst mode select Jumper not installed: selects normal refresh (non-burst) Jumper on pins 1 and 2: Normal refresh selected Jumper on pins 2 and 3: four burst refresh selected ¹

NOTES:

1. Factory setting.
2. With the jumper in this position the hardware Power Fail check is disabled.
3. Jumpers J303, J306, and J307 must all be set for the same pin pairs, i.e. pins 1 and 2 or 2 and 3.

P301 on the I/O card is the connector which supplies control signals for the key panel on the front of the computer. Figure 4-4 describes the signals at this connector.

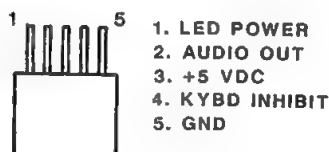


Figure 4-4. P301 Connector Signals

Memory

Each system memory card contains one 8-section switch for configuration purposes. Figure 4-5 illustrates a typical system memory card. Figure 4-6 illustrates the factory default settings for switch SW401.

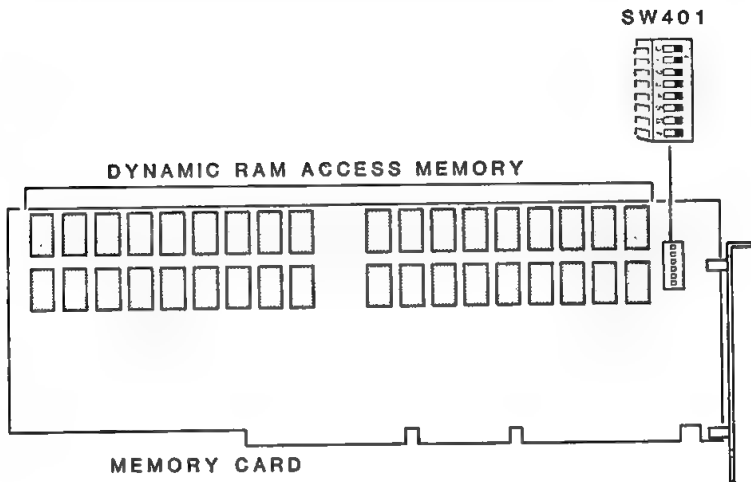


Figure 4-5. System Memory Card

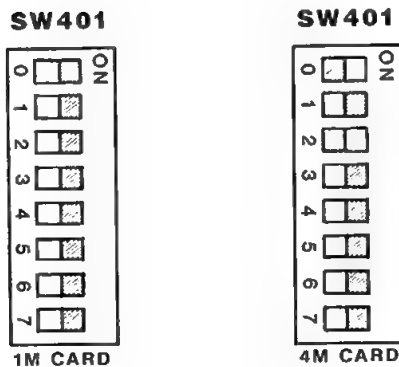


Figure 4-6. SW401 Default Settings

Hardware Configuration

A maximum of four memory cards may reside in the system at one time. Switch sections 0 and 1 provide the means for identifying each card in the system. Figure 4-7 describes the various switch settings.

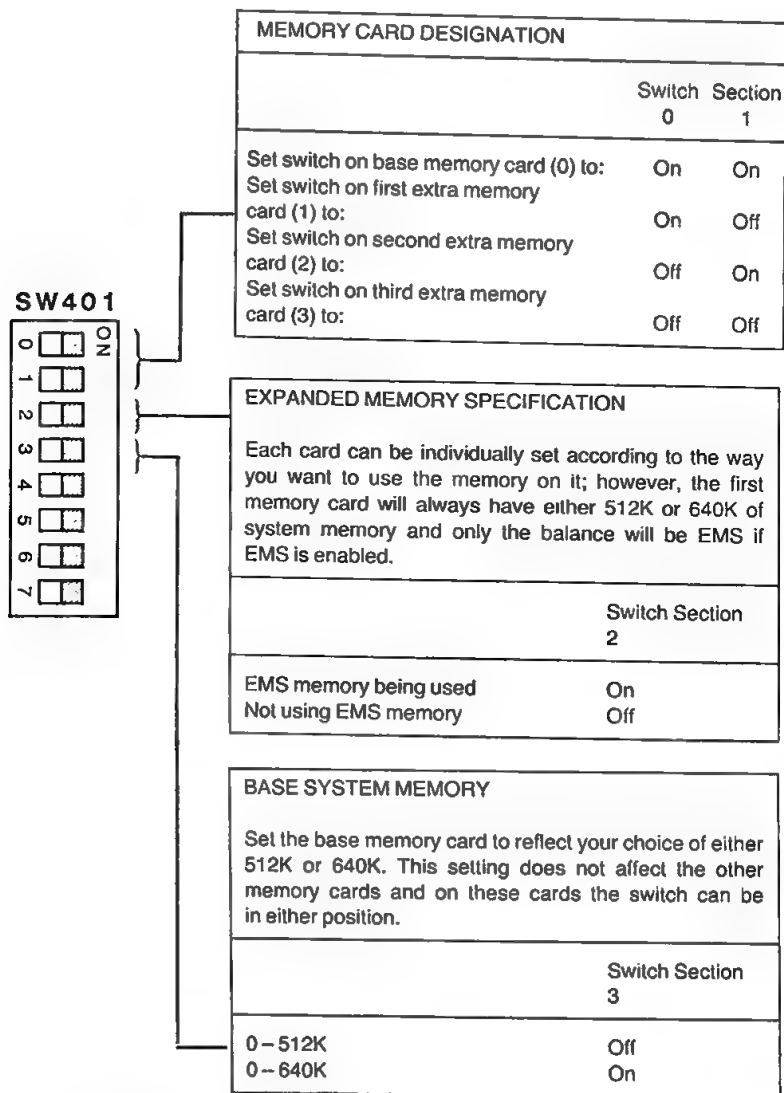


Figure 4-7. Memory Configuration Options

Hardware Configuration

MEMORY CARD BASE ADDRESS				
In the table below, sections divided by:				
----- denotes a 2-megabyte boundary				
_____ denotes a 4-megabyte boundary				
To determine which setting to use, refer to the explanation which follows this table.				
Megabyte Boundary	Switch Section			
	4	5	6	7

0	On	On	On	On
1	Off	On	On	On

2	On	Off	On	On
3	Off	Off	On	On

4	On	On	Off	On
5	Off	On	Off	On

6	On	Off	Off	On
7	Off	Off	Off	On

8	On	On	On	Off
9	Off	On	On	Off

10	On	Off	On	Off
11	Off	Off	On	Off

12	On	On	Off	Off
13	Off	On	Off	Off

14	On	Off	Off	Off
15	Off	Off	Off	Off

SW401

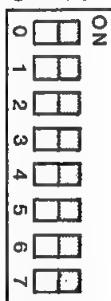


Figure 4-7 (continued). Memory Configuration Options

Hardware Configuration

Section 2 of switch SW401 determines whether to use part of the available memory as EMS memory. Placing this switch section in the ON position allows the system to use up to a maximum of 2M per card as EMS memory. How much memory is allocated depends on the type of memory card installed in the system:

- If you are using a 1M memory card as the system memory card, approximately 256K can be used as EMS memory.
- If you limit the size of the base system memory to 512K, the amount of EMS memory will increase to approximately 384K.
- If you are using a 1M memory card as expansion memory, the entire 1M of RAM on the card will be assigned as EMS memory.
- On a 4M memory card, the system will reserve 2M of the available memory as EMS memory. This allocation will occur regardless of whether you use the card as the base memory card or as expansion memory.

If the EMS option is disabled by placing the switch in the OFF position, all memory above 1M boundary becomes extended memory. Extended memory is normally used by RAM disk-type utilities to create virtual drives within system memory. The system base memory is not affected by the use of EMS memory; the base memory card always retains a specific amount of memory for the system memory.

Section 3 of SW401 determines how much memory on the base memory card is system memory. If the switch is in the OFF position, 512K becomes system memory space. If the switch is in the ON position, 640K becomes system memory space.

The remaining switches on SW401 determine the base address for each board. New memory cards must be installed on 1M memory boundaries. Refer to Figure 4-7 for the switch settings and the memory boundaries.

System Cache Card

The optional system cache card provides increased system throughput whenever the CPU must access system memory. The card is designed to be transparent to the system as well as to the user. The design of the card is fixed and there are no user-selectable options.

Floppy/Winchester Controller Card

The controller card connects the floppy and hard disk drives with the rest of the computer. The controller card controls the transfer of data to and from the drives. The controller card supports up to two floppy disk drives (either 360K or 1.2M) and two hard disk drives (either 40M or 80M). The jumpers on the controller card are preset by the manufacturer and should not be changed. Figure 4-8 illustrates the controller card and jumpers.

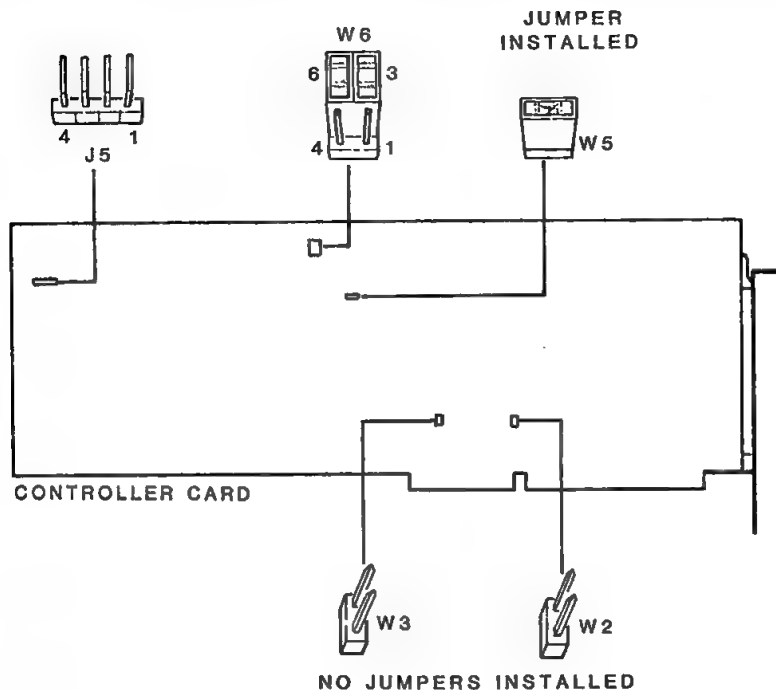


Figure 4-8. Drive Controller Card

Disk Drives

This computer system can accommodate several drive types. Most drives allow the user to set the device number on the drive. This is done so that systems using more than one drive can correctly address the appropriate drive during read and write operations. If a second drive is added to the system, its drive address is normally selected via an address selection jumper on the drive.

In IBM-PC computer systems, and most compatibles, including this computer, that is not the case. In this computer, drive addressing is accomplished using a special wiring arrangement within the disk drive cables. The drive address jumpers are set to the same address for all similar drives in the system. If you add additional drives to the system, make sure that the address jumpers are set properly. All drives in this computer system are addressed as drive unit number one. The following sections describe these configuration options.

Floppy Drives

Three different types of floppy disk drives can be installed in this computer. A number of different manufacturers supply the various drive types. The following figures illustrate the most commonly used drives.

A slight modification may be required if you are using a standard 5.25-inch 360K (Z-207-7) floppy drive. If you install this type of drive as the second drive in the computer, you will need to open the signal line at pin 34 of the drive connector. Figure 4-9 illustrates this modification. Select the illustration which most closely resembles the drive you have and make the modification.

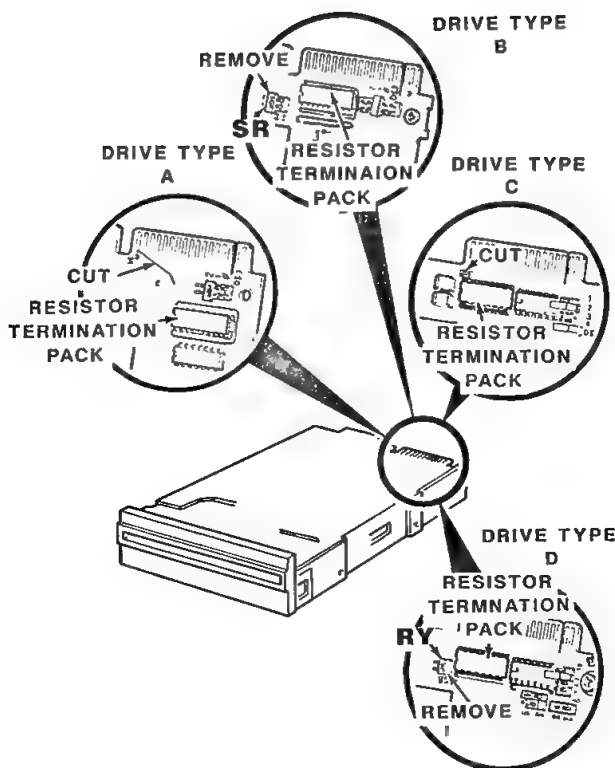


Figure 4-9. Optional Drive Modification

Hardware Configuration

Figure 4-10 illustrates the address jumper location on a typical 5.25-inch, 360K floppy disk drive.

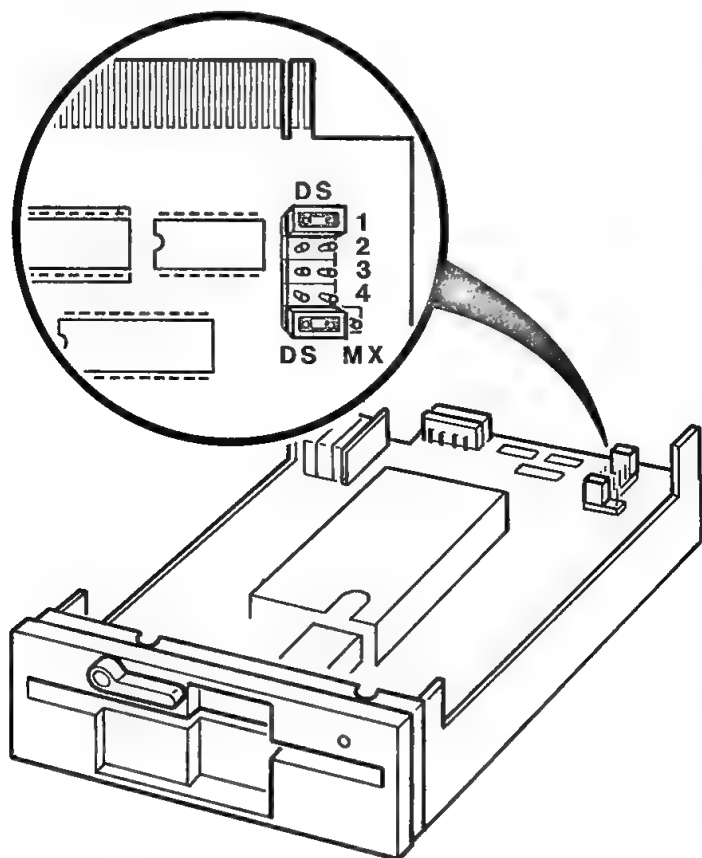


Figure 4-10. 5.25-Inch 360K Drive

Figure 4-11 illustrates the address jumper location of a typical 5.25-inch, 1.2M floppy disk drive.

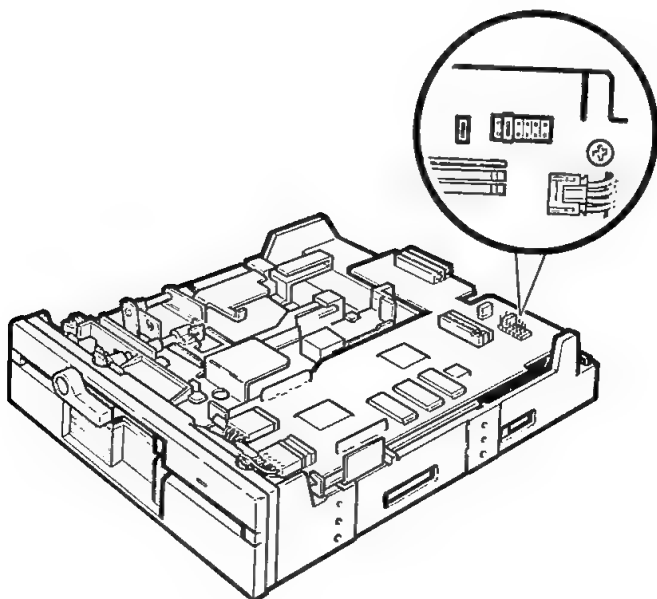


Figure 4-11. 5.25-Inch 1.2M Drive

Hardware Configuration

Figure 4-12 illustrates the address jumper location on a typical 3.25-inch, 720K or 1.4M micro-floppy disk drive.

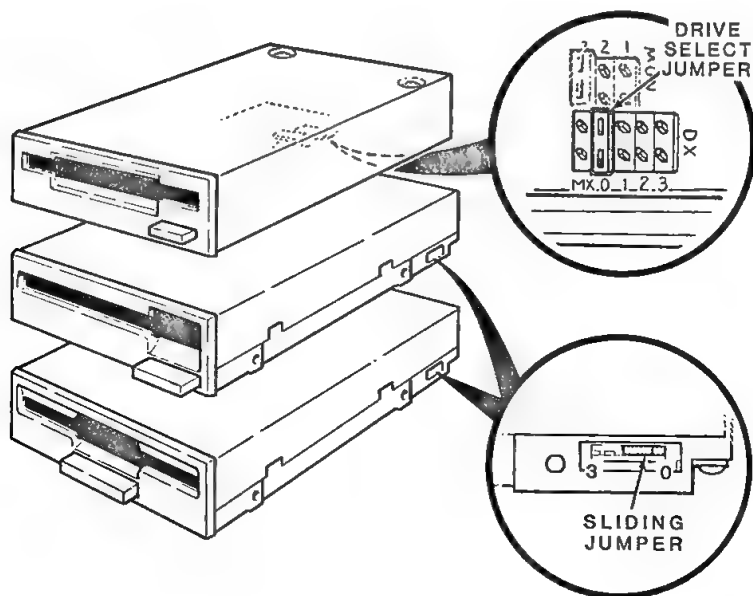


Figure 4-12. 3.25-Inch 720K Drive

NOTE: The resistor terminating pack must be moved to the last drive on the cable when adding additional drives.

Each floppy disk drive is equipped with a number of jumpers that are used to configure different operating parameters for the drive. Specific information on these jumpers can usually be obtained from the data sheets published by the drive's manufacturer. Figures 4-13 through 4-16 illustrate the different types of circuit cards found on the floppy drives and the configuration jumper settings for each. Locate the figure that most closely resembles the drive you are using and verify that all jumpers are in the indicated positions.

Hardware Configuration

NOTE: The configuration settings used in Figures 4-13 through 4-16 are only valid for 5.25-inch form factor drives. If you are using a 3.5-inch form factor drive, refer to the instructions that came with it.

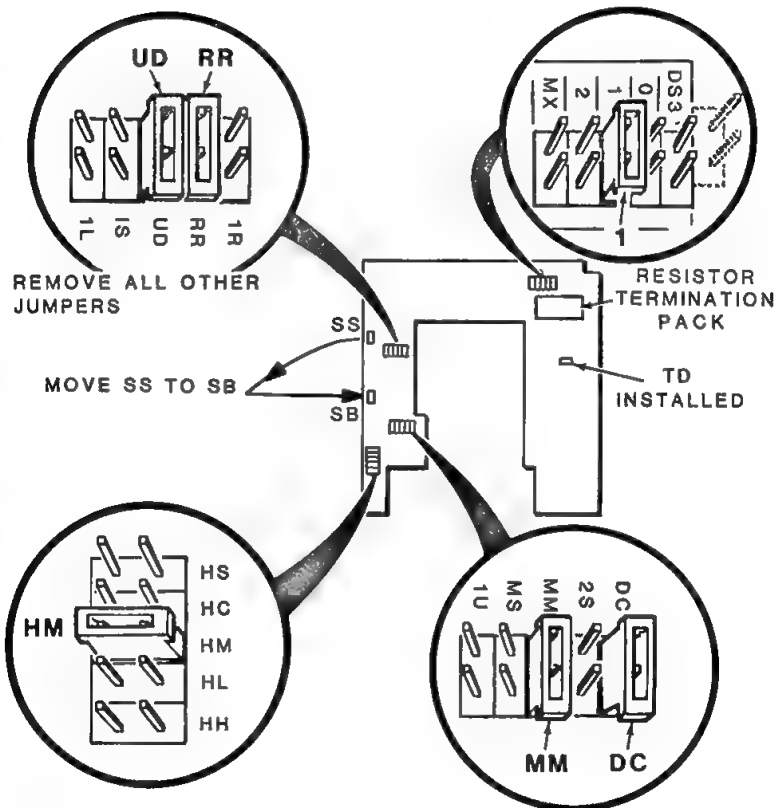


Figure 4-13. Drive Type A Configuration Settings

Hardware Configuration

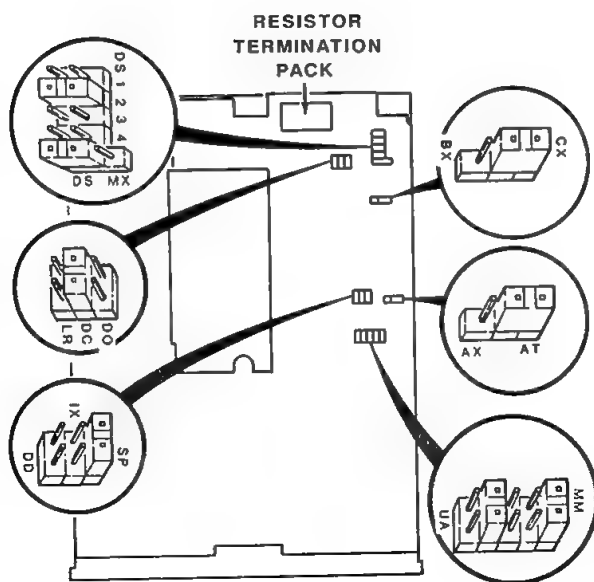


Figure 4-14. Drive Type B Configuration Settings

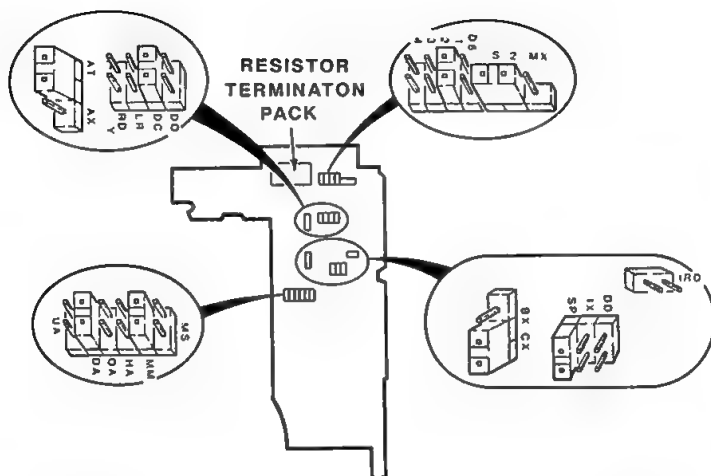


Figure 4-15. Drive Type C Configuration Settings

Hardware Configuration

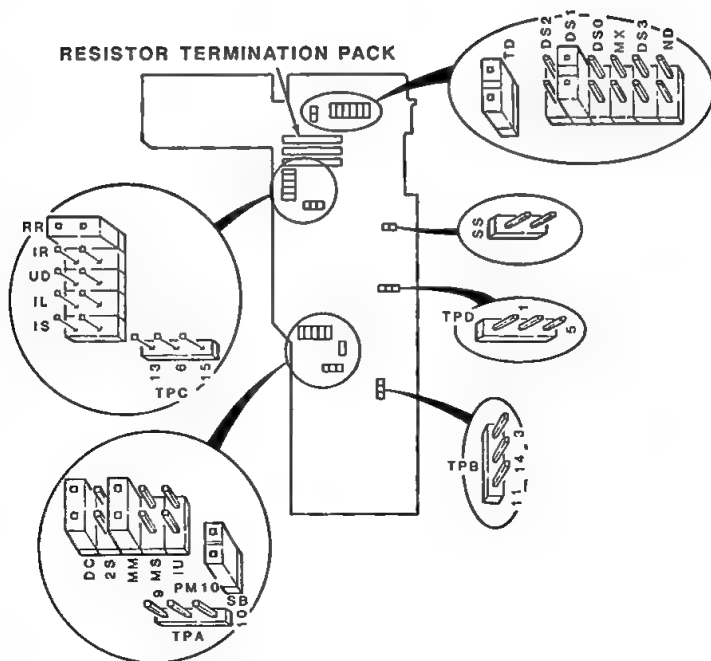


Figure 4-16. Drive Type D Configuration Settings

Fixed Hard Disk Drives

Hard disk drives, like floppy disk drives, come in a variety of sizes and data storage capacities. The same addressing procedure is used as with floppy disk drives: the drive cables are modified to properly set the drive addresses.

Figure 4-17 illustrates a typical 5.25-inch, half-height, 20 megabyte hard disk drive. The figure shows the location of the drive address jumpers and the resistor terminating pack. As with floppy drives, the terminating pack must be moved to the last drive on the cable.

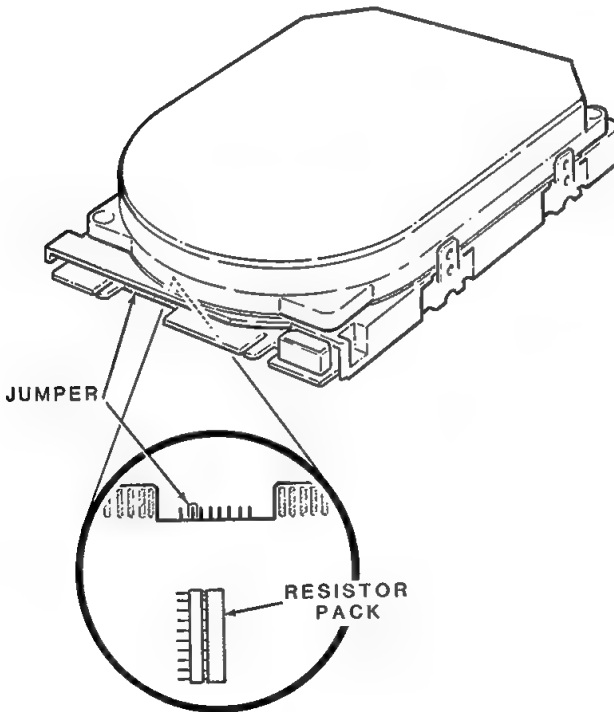


Figure 4-17. 5.25-Inch 20M Drive

Figure 4-18 illustrates a typical 5.25-inch, full-height, 20 or 40 mega-byte hard disk drive.

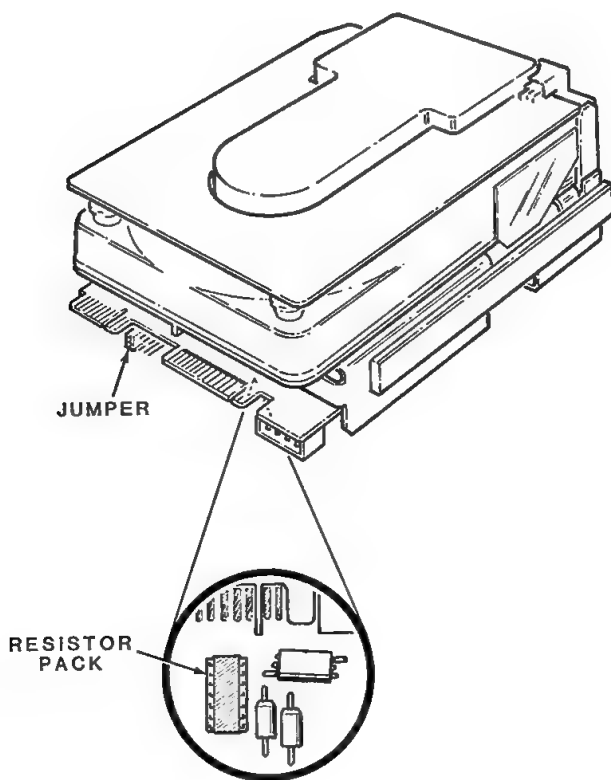


Figure 4-18. 5.25-Inch 20M/40M Drive

Hardware Configuration

Figure 4-19 illustrates a typical 5.25-inch, full-height, 40- or 80-megabyte hard disk drive.

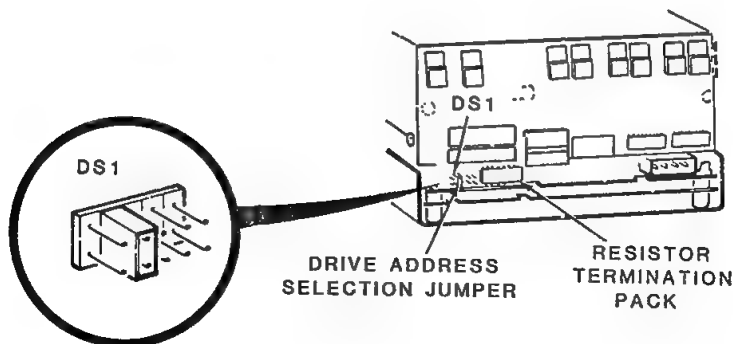


Figure 4-19. 5.25-Inch 40/80M Drive

Removable Cartridge Hard Disk Drives

Removable cartridge hard disk drives combine the features of floppy and hard disk drives. Like a floppy disk drive, the media may be removed from the drive. Like a hard disk drive, the total storage capacity of the drive is much larger than on a standard floppy disk. For all practical purposes, though, these drives should be considered the same as standard hard disk drives. Figure 4-20 illustrates a typical removable cartridge hard disk drive.

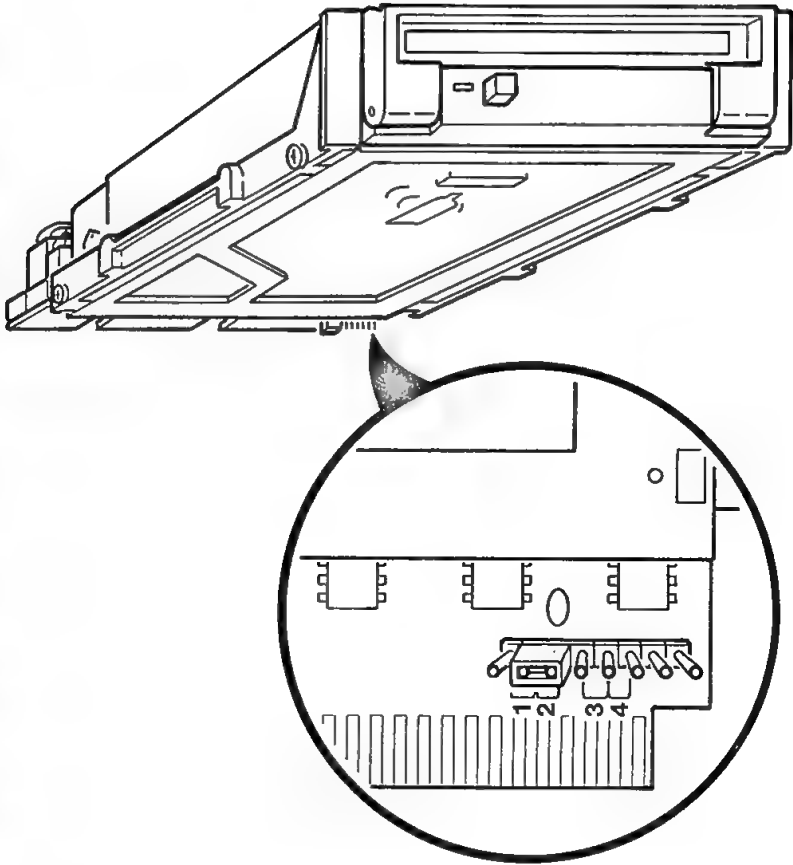


Figure 4-20. Removable Cartridge Drive

NOTE: When installing an additional hard disk drive it may be necessary to run the PREP and PART utilities in the operating system to prepare the drive for use. Refer to your operating system manual for the correct procedures.

Hardware Configuration

Video System

This computer comes equipped with a 31 kHz enhanced video display card. This video card supports the common PC-compatible display standards by emulating the Color Graphics Adapter (CGA), the Enhanced Graphics Adapter (EGA), the Monochrome Display Adapter (MDA) and the Hercules Graphics Card (HGC). In addition to the usual modes, this card provides a 640×480 enhanced video mode. Two independent output connectors are available, one for 31 kHz analog display systems, the other for the usual EGA, CGA, and TTL monochrome displays. Figure 4-21 illustrates the video card and the switch and jumper locations. The connector pinouts for the video card are described in Chapter 2 of this manual.

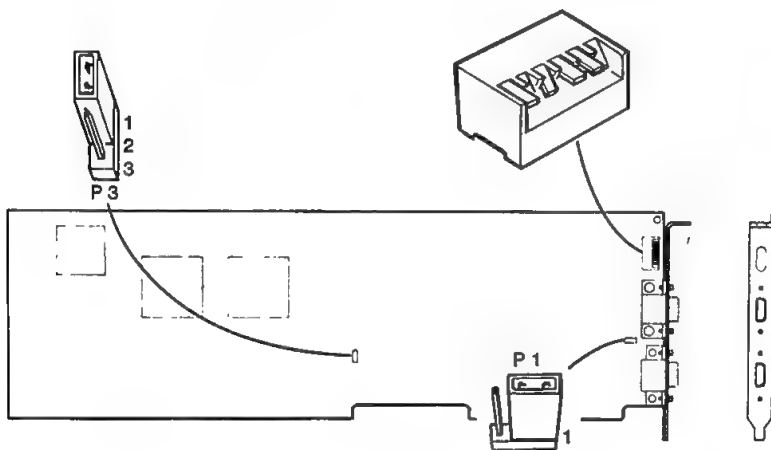


Figure 4-21. 31 kHz Video Switches and Jumpers

Hardware Configuration

CAUTION

The video card in this computer can only be used with either TTL- or analog-level display monitors. If section 6 of the video configuration switch is in the OFF position, only TTL-level monitors should be connected to the card using the 9-pin connector. If section 6 is in the ON position, only monitors that support 31 kHz analog signals should be connected to the 15-pin connector. **DO NOT** leave TTL-level monitors connected to the 9-pin connector during 31 kHz operation. Use of any other type of monitor or incorrect use of the monitor may result in damage to the monitor, the card, or both.

The 6-section DIP switch, in conjunction with jumpers P1 and P3, determines the mode of operation selected for the card. Table 4-3 describes the available options. Only switch sections 1 – 4 are necessary for configuration of the basic monitor modes.

The card comes pre-configured to drive either the ZCM-1390 or the ZCM-1490 monitors. If you are using either of these monitors, no action is necessary.






















Switch section 5 sets automode on or off. Automode will automatically switch the display output of the video card between CGA or EGA modes. This mode will also automatically switch between MDA and HGC modes on a monochrome monitor.

Switch section 6 enables the 31 kHz display mode of this video card. When this mode is enabled, only an analog RGB monitor with a 31 kHz bandwidth can be used with the card. A separate 15-pin output connector is supplied for this type of monitor. If this mode is enabled while a standard or monochrome monitor is connected to the 9-pin connector, damage could result to the monitor, the card, or both.

The video card can display a 3-, 14-, or 16-level gray scale on a monochrome monitor. Using an enhanced color monitor, the card can display up to 16 colors at a time from a palette of 64 different colors.

Hardware Configuration

Table 4-3. Video Option Selections

MONITOR	DISPLAY	CONNECTOR	SWITCH JUMPERS		
			SW1	P1	P3
ZVM-1330 or other CGA monitor	CGA (15.70 kHz) (Color Graphics Adapter)	9-pin	 OFF ON ON OFF OFF OFF ON	 3 2 1	 3 2 1
ZVM-1380-C or other EGA monitor	EGA (Enhanced Video RGBI/ RGBrgb)	9-pin	 OFF ON OFF ON ON OFF ON	 3 2 1	 3 2 1
ZVM-1470-G or other EGA monitor	EGA (Enhanced Video RGBI/ RGBrgb)	9-pin	 OFF ON OFF ON ON ON OFF	 3 2 1	 3 2 1
ZVM-1240 or other MDA monitor	MDA (TTL monochrome display)	9-pin	 OFF ON OFF ON OFF OFF OFF	 3 2 1	 3 2 1
ZCM-1390* or other VGA monitor	480 Line VGA-like display	15-pin	 ON ON OFF ON ON ON OFF ON ON OFF	 3 2 1	 3 2 1
ZCM-1490* or other VGA monitor	480 Line VGA-like display	15-pin	 ON ON OFF ON ON ON OFF ON ON OFF	 3 2 1	 3 2 1
ZMM-149 or other monochrome monitor	480 Line VGA-like monochrome display	15-pin	 ON ON OFF OFF OFF ON ON ON ON	 3 2 1	 3 2 1

*Factory settings

Video Drivers

A number of software applications can take advantage of the enhanced performance of this video card by using a video driver program. A number of driver programs are provided with this computer system. Along with the programs, which are contained on a disk, are instructions describing the programs and how to use them. Please refer to these instructions before attempting to use the video drivers with any of your software.



Chapter 5

Operation

This chapter describes the keyboard functions, how to use the firmware-driven Setup/Configuration program, and how to get the system booted. Information about the MS-DOS operating system is not covered in this chapter. If operating system information is required, refer to the MS-DOS user and reference guides supplied with your computer.

Keyboard

The keyboard features 101 keys that provide alphanumeric and control operations. Alphabetic keys are arranged in the standard QWERTY layout. Numeric keys are positioned in a separate keypad on the right side of the keyboard and along the top of the alphabetic key group.

Control keys include the cursor keys, the screen control keys, and other special keys. There are also 12 function keys along the top of the keyboard that can affect system control. Figure 5-1 illustrates the layout of this keyboard.

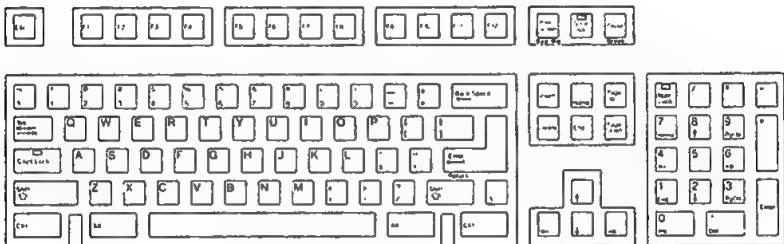


Figure 5-1. 101-Key Keyboard

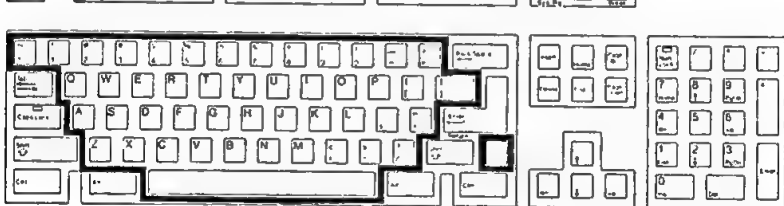
All keys, except the SHIFT keys, CTRL (control) and ALT (alternate)

NOTE: You may encounter some software programs that do not

The following section provides a brief description of the al-

Alphanumeric — This block of keys contains the alphabetic charac-

100



Enter/Return — The ENTER/RETURN key is used to return the cursor to the left side of the display (software usually moves the cursor to the next line). Also, pressing the ENTER/RETURN key after data or instructions have been entered tells the computer to process them.

Tab — Pressing this key moves the cursor to the next tab setting.

Space bar — Pressing the space bar enters a blank character (space).

Back space — The BACK SPACE key moves the cursor back one space. In many software programs, pressing the BACK SPACE key also erases characters as it moves the cursor to the left.

Shift — There are two SHIFT keys on the keyboard, one on the right side and one on the left. Normally, when a SHIFT key is pressed, capital letters, symbols, and extra punctuation marks are generated. However, if the CAPS LOCK key is active, pressing the SHIFT key causes the letter keys to generate lowercase letters. The left and right SHIFT keys generate different codes. Some programs may be written to specifically recognize either key.

Ctrl (Control) — There are two CTRL keys, one on the right side of the keyboard and one on the left. The CTRL keys are the most commonly used method for entering system commands. Normally, the CTRL key is pressed and held, and then another key or keys are pressed. The CTRL key is often symbolized by a caret (^). The left and right CTRL keys generate different codes. Some programs may be written to specifically recognize either key.

Some of the commonly used CTRL key combinations are:

CTRL-ALT-DEL — When the CTRL key and ALT key are pressed and held and the DEL (or DELETE) key is then pressed, the computer executes the same initialization process as when it is first turned on. This is also known as resetting or warm booting.

Operation

CTRL-ALT-RETURN/ENTER — This key combination is similar to CTRL-ALT-DEL except this sequence allows the user to return to the program just exited. This feature is useful for running the Monitor-based debugger when developing software. Using this sequence can result in problems; refer to Chapter 6 for a detailed discussion.

CTRL-ALT-INS — When the CTRL and ALT keys are pressed and held and the INS (or INSERT) key is pressed, the computer branches to the Monitor program. Since the computer is not actually reset, you can boot from an alternate drive using Monitor commands, even though the computer is set to autoboot.

CTRL-S — When these keys are pressed, output on the display pauses until you press another key. This feature is useful when scrolling large text files. To resume the scrolling action, press the CTRL-Q key sequence or any printable character key.

CTRL-NUM LOCK — This key combination halts the computer until you press another key.

CTRL-SCROLL LOCK — When these keys are pressed, any Monitor program command in progress ends and the computer returns to the Monitor prompt (->).

Alt (Alternate) — There are two ALT keys, one on the right side of the keyboard and the other on the left. The ALT keys are similar in operation to the CTRL keys and are used the same way for entering commands; however, they are used far less frequently. The left and right ALT keys generate different codes. Some programs may be written to specifically recognize either key.

NOTE: The SHIFT, CTRL, and ALT keys are always used with other keys. They perform no function when used alone. The side-by-side arrangement of the CTRL and ALT keys on both the right and left side of the keyboard makes it convenient to use these keys in combination with other keys.

The cursor control keys and the calculator keypad make working with spreadsheet software programs easier. For example, the separate cursor control and keypad sections allow for efficient entry of numeric data.

There are two sets of HOME, END, PAGE UP (PGUP), PAGE DOWN (PGDN), and arrow (cursor control) keys. One set is on the factored screen control key section; these key functions are always active. The other is in the calculator keypad section and these functions are only active when the NUM LOCK key is inactive (or the SHIFT key is active while the NUM LOCK key is active). Figure 5-3 illustrates these keys.

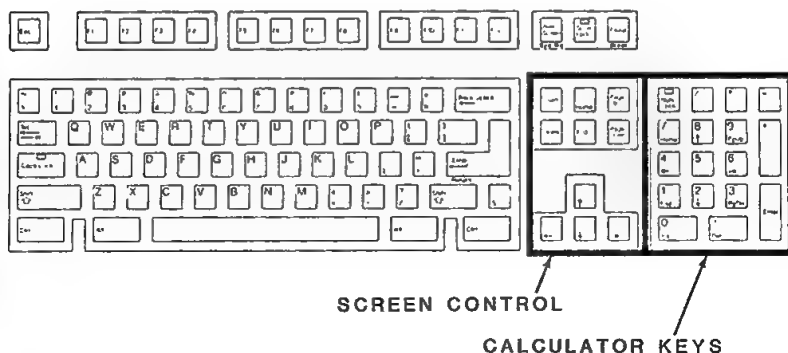


Figure 5-3. Screen Control and Calculator Keypad Keys

Some software programs may not allow all functions to be performed using the separate arrow keys and the keys immediately above them (INSERT, DELETE, HOME, PAGE UP, etc.). In these cases, it may be necessary to use the corresponding keys in the calculator keypad. These keys are defined in the following section.

Arrows — The two sets of arrow keys indicate the direction in which they move the cursor in most applications programs and in the operating system. The arrow keys make it easy to position text in word processing programs or to quickly enter cell values in spreadsheet programs.

Home — In many programs, pressing the HOME key moves the cursor to the upper-left corner of the screen. With spreadsheet programs, pressing the HOME key usually moves the active cell indicator to the upper-left corner of the spreadsheet.

Operation

End — The END key is typically used by applications programs to move the cursor to the lower-right corner of the screen display. In some spreadsheet programs, pressing the END key moves the cell indicator to the most remote cell.

Page Up/Pg Up and Page Down/Pg Dn — These keys are used by many programs to move the cursor up or down a certain number of lines.

Insert (Ins) — These keys allow you to enter the insert mode. In certain word processing programs, the insert mode determines whether text you type pushes existing text aside to make room (on), or simply types over and replaces existing text (off). To enter the insert mode, press the INSERT (or INS) key; to exit the insert mode, press the key again. When used simultaneously with the CTRL and ALT keys, the INSERT (or INS) key can also be used to return the computer to the Monitor prompt.

Delete (Del) — In some application programs, you can backspace and erase the character to the left of the cursor by pressing either of these keys. When used simultaneously with the CTRL and ALT keys, DELETE (or DEL) is used to reset the computer.

Num Lock (Number Lock) — When the computer is turned on, the LED on this key is lit. It indicates that the number keys and decimal point on the calculator keypad are active. However, if either SHIFT key is pressed when NUM LOCK is active, the number keys will become inactive. To turn this feature off (and on again), press the NUM LOCK key. The LED will go out and the cursor control functions and INS and DEL keys will be active.

There are 12 function keys for increased automatic operation, three dedicated function control keys, and an ESC (escape) key along the top row of the keyboard. The following paragraphs describe these keys and Figure 5-4 illustrates their location.

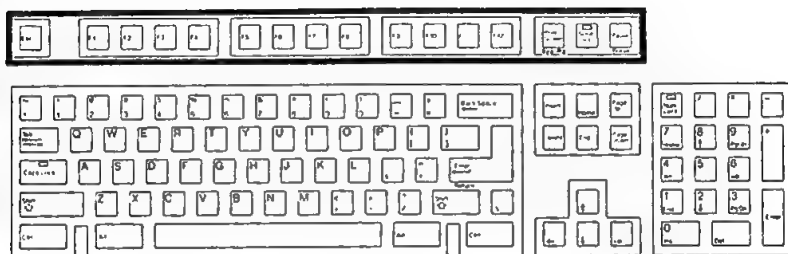


Figure 5-4. Function Keys

Esc (Escape) — The ESC key is typically used to discontinue program execution. It is isolated from the other keys so it is less likely to be inadvertently pressed. The ESC key also performs special duties when used in combination with another key in some applications.

F1 through F12 — The function keys are used for special purposes by some programs. For example, in most word processing programs, the function keys perform such functions as indenting, setting right and left margins, underlining, and boldfacing type.

The function keys have special capabilities in the MS-DOS operating system. For example, function key F3 allows you to reuse all or part of a previously-entered command. (For further information, refer to the *MS-DOS User's Guide*.)

NOTE: In some older software programs, the F11 and F12 keys may not perform any function.

Print Screen/System Request — When this key is pressed all ASCII information appearing on the screen is sent to the printer. This function only operates when the system is in the MS-DOS command mode. The printer must be on, on line, and have paper ready. If not, the computer pauses until the printer is made ready. If you want to print graphics or special characters, refer to the Print Screen command in the *MS-DOS User's Reference*. SYSTEM REQUEST is used in conjunction with the ALT key and is similar to the BREAK key. It is usually defined by the program that is using it.

Operation

Scroll Lock — In some spreadsheet programs, this feature allows the cursor to remain stationary while the screen scrolls. When inactive, the cursor can be moved while the screen remains stationary. To activate this feature, press the SCROLL LOCK key. The LED indicator will light. To cancel the feature, press the key again. When the SCROLL LOCK key is used with the ALT key, it empties the system's type-ahead buffer.

Pause/Break — When this key is pressed, it freezes the display of text on the screen. If you enter an MS-DOS TYPE command, for example, the text of the file rolls up the screen faster than you can read it. Pressing the PAUSE key will stop it temporarily. Pressing any other key will allow it to resume.

AT/XT Compatibility

This advanced keyboard operates with AT-compatible or XT-compatible computers by simply changing the position of a switch located under the keyboard nameplate. In this computer, the keyboard is configured for operation as an AT-compatible keyboard. DO NOT switch the keyboard to the XT mode while it is being used with this computer.

You may, however, use the keyboard with an XT-compatible computer by switching to the XT mode. To do this, gently remove the keyboard nameplate using a flat-bladed screwdriver or similar tool. Place the switch in the XT position and replace the nameplate. Refer to Figure 5-5.

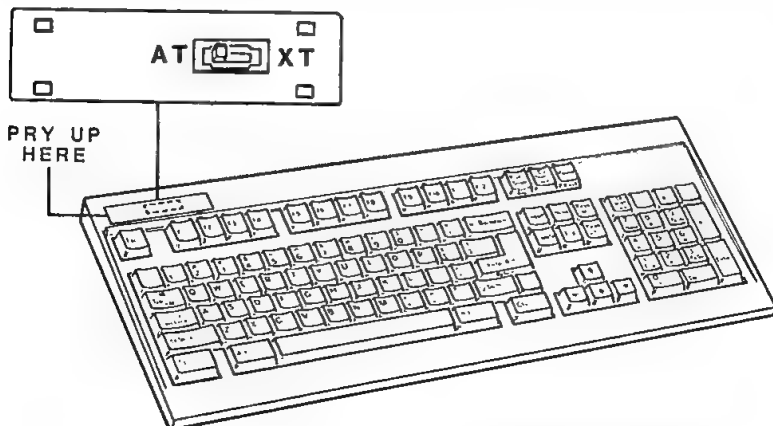


Figure 5-5. AT/XT Compatibility Switch

Keyboard Adjustment

You can adjust the tilt of the keyboard so that it is at the most comfortable angle for you. To increase the tilt, raise the feet on the bottom of the keyboard. To decrease the tilt, place the feet in their closed position. Refer to Figure 5-6.

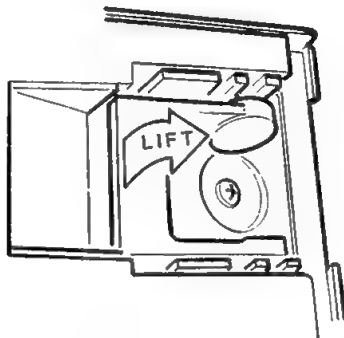


Figure 5-6. Keyboard Adjustment

Using the Setup/Configuration Program

The Setup/Configuration program reports the system configuration to the computer's CPU. You can set the time and date, enter the amount of memory installed in the computer, choose the boot drive, enter the type of video card, and determine the number and type of floppy and hard disk drives installed in the system. A special entry will slow down the speed of the computer as required for time dependent applications. All of the entries are made from the keyboard instead of setting hardware switches and installing jumpers.

Once information is entered in the Setup/Configuration program, it is retained in non-volatile memory. If you change the configuration of the computer or if you replace the battery on the backplane, you must update the Setup/Configuration program.

To access the Setup/Configuration program:

1. Press and hold the CTRL and ALT keys and then press the INS (or INSERT) key. After a short delay, the Monitor program prompt (->) will appear on the screen.

Operation

2. Type **SETUP** and press the **ENTER/RETURN** key. You will see a display similar to the one illustrated in Figure 5-7.

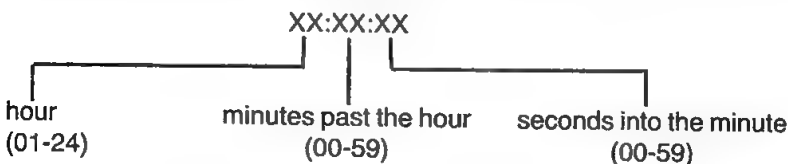
Z-300 Hardware Setup/Configuration Program			
Time:	01:21:50	DST:	Enabled
Date:	01/01/1987		Disabled
Base Memory Size:	640K bytes		
Expansion Memory:	1024K bytes		
Floppy Drive 0:	1.2M	None	
Floppy Drive 1:	1.2M	None	
Boot Drive:	Floppy Drive 0 Hard Disk Drive 0 Floppy then Hard Disk Enter MFM-300 Monitor		
Video Display:	Color Card: 40x25 Color Card: 80x25 Mono. Card: 80x25 Enhanced Graphics		
Video Refresh Rate:	50 Hz	60 Hz	
Operating Speed: Slow Fast Smart Hard Disk Drive 0: Drive Type 2 Media Type: Fixed Cylinders: 615 Heads: 4 Ship Zone: 615 Sectors: 17 Precomp: 300 Capacity: 21M Hard Disk Drive 1: -Not Present- Media Type: N/A Cylinders: Heads: Ship Zone: Sectors: Precomp: Capacity:			
Enter Current Time As HH:MM:SS In 24 Hour Format Use Space/Backspace to select values, Arrows to move, Esc when done			

Figure 5-7. Setup Menu

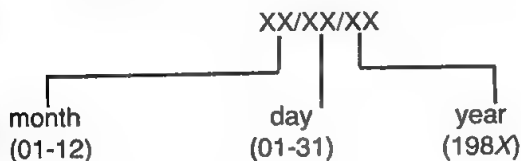
The reverse-video images you see on the display are the factory selections in each "field", or group of options. The directions you need to proceed through the program are located at the bottom of the display. These directions change as you go through the program.

When you enter the time and date, use the number keys to type in the time and date the Enter/Return key to lock in the selection. If you happened to enter incorrect information, use the **BACKSPACE** key to return to the position where you made the mistake, then type over it.

Time — The time is entered in a 24-hour format. (it is not necessary to type the colons, they are entered automatically):



Date — The date is entered in the following format (it is not necessary to type the slashes, they are entered automatically):



In the remaining fields, use:

- The arrow keys to move from one field to another.
- The space bar and BACKSPACE keys to make a selection in the field. The selection is activated when you move to another field.
- The ESC key to exit from the program when all information has been correctly entered.

Base Memory Size — Choose the size of the base memory in the computer. If you increase the amount of base memory, change this entry accordingly.

Expansion Memory — If this entry reads "0K Bytes" (zero kilobytes) it indicates base memory only. Whenever expansion memory is added to the system change this entry accordingly. The selections range from 0K to 15360K in increments of 64K.

Floppy Drive 0 — Indicate the size of the first floppy drive in this field. This computer is shipped with a 1.2M drive as the first drive in the system. If a different type of drive is used, enter the correct data in this field.

Floppy Drive 1 — If you add a second floppy drive to the system, enter the size of the drive in this field.

Boot Drive — Select the drive that you want the computer to auto-boot from. There are four choices: boot from floppy; boot from hard disk; boot from floppy if present, if not, boot from hard disk; or boot to the Monitor program. If you choose to have the computer access its Monitor program you can manually boot to any valid drive (A, B, C, etc.)

Operation

Video Display — Select the entry that describes the character display of the video card in the computer. If an optional high-resolution monochrome display adapter is installed in the computer and you plan to use it as the default display, select "Mono. Card".

Video Refresh Rate — Select the entry that describes the power line frequency in your area. In the United States, it is typically 60 Hz. If the setting is incorrect, the video display will flicker.

Operating Speed — Select the entry that describes the speed characteristics you wish to use. Normally, the fast mode is selected. However, some programs may require a slower input/output speed in order to run. The Smart mode slows the computer down for the input/output operations, but allows the fast speed for other operations. If the program still does not run properly, it may require the slow mode. This mode slows down the complete operation of the computer and may be necessary for some programs.

Hard Disk Drive 0 — Depending upon the model you have, your system has either a 20-, 40-, or 80-megabyte hard disk drive. This section of the Setup/Configuration program menu reflects the specifications of various drive types. The window below the highlighted drive type lists the drive parameters. The capacity entry in the window displays the actual formatted capacity of the drive, even though it is often referred to as either a 20-, 40-, or 80-megabyte drive.

Hard Disk Drive 1 — If you install a second hard disk drive, refer to Table 5-1. This table lists a number of different drives and the corresponding type number. When you install a drive, the type selection must match the parameters of your drive. You may review the different drive types available by pressing the space bar or the BACK SPACE key. Pressing the ENTER key will select the highlighted option.

Table 5-1. Hard Disk Drive Types

CAPACITY	MANUFACTURER	MODEL	DRIVE SIZE
10M	MiniScribe	2012	1
10M	CMI	5412	1
10M	Seagate	412	1
20M	Seagate	ST-225	2
20M	Seagate	4026	2
20M	CDC	94155-28	2
20M	Seagate	ST-225-2	2
20M	Tandon	TM-262	33
20M	MiniScribe	8425	34
20M	Lapine	TITAN 3532	15
30M	Hitachi	DK511-3	3
40M	Hitachi	DK511-5	7
40M	CDC	94155-48	10
40M	Seagate	ST-4051	16
40M	CDC	94205-51	43
80m	CDC	94155-86	39
80M	MiniScribe	6085	40
80M	MicroScience	HA-1050	41

When you enter the drive type number, the information about your drive will be shown on the screen. If you install a drive that is not listed in the table, you will have to step through the drive types on the menu until the displayed information matches the specifications of your drive.

When you have completed all the changes to the Setup/Configuration program and it accurately reflects your system, press the ESC key. The computer will display the following prompt:

Are You Done Making Changes <Y/N>?

If you are satisfied with your selections, press the Y key for "yes" and then press ENTER/RETURN. Your entries will be saved and the computer will perform the boot-up procedure you selected.

If you do not wish to save the changes you made, press the N key for "no" and then press ENTER/RETURN. The original entries will remain in effect. Press the ESC key to return to the Monitor prompt (->).

System Boot Procedure

When this computer is shipped from the factory, the Setup/Configuration program is set to boot from the hard disk drive. However, there is no program on the hard disk. Since the attempted boot procedure fails, the following error message will appear on the video display:

Partition not found

This message simply means that the boot procedure failed because no program information was available on the hard disk. At this point the system is halted.

To continue, press the CTRL-ALT-INS key combination. The computer will enter the Monitor program and display the ROM version number and Monitor prompt (->). At this point, refer to your MS-DOS documentation and place disk 1 in floppy drive A. Be sure to close the disk drive latch or the drive will not respond to system commands. To boot from the floppy disk, type `bf` and press ENTER/RETURN. The computer will access floppy drive A and attempt to read the operating system program from the disk.

After the program has been read, the video monitor will display some type of opening message or prompt. For example, the MS-DOS operating system will display its version number, prompt you to enter the correct date and time, and then display its operating system prompt (`A>`). to use the operating system, follow the instructions in the operating system manual. These directions will help you configure the computer to your requirements.



Part II

System Programming



The Monitor Program and Programming with Interrupts

This chapter provides an overview of the Monitor program and its operation. Each time the computer is turned on a number of system interrupts become available for use. Chapters 7 through 11 explain each interrupt.

The Monitor Program

The Monitor program, MFM-300, maintains a high level of software compatibility with desktop PC-compatible computers. This is largely done through the use of firmware-established interrupts for most control and input/output routines. The Monitor program also contains self-tests that are performed each time the computer is turned on. Additional routines provided in the Monitor program are user-executed tests, video commands, disk boot routines, and a machine language debugger. The program also controls the Setup/Configuration program. Chapter 5 contains more information about the Setup/Configuration program.

The Monitor program does not use the full capabilities and power of the 80386. In order to maintain compatibility with earlier Zenith hardware and firmware releases, MFM-300 operates identically to earlier firmware. This means that the Monitor program treats the 80386 as if it were an 8088 and all registers and status flags reported by the program are as they appear in an 8088. Since the 80386 duplicates these registers and flags, the reported information is correct. However, the 80386 also has extended register and flag capabilities that the Monitor program does not support. This reduces the usefulness of the Monitor program, but it does provide a user interface consistent with earlier systems. This chapter covers only the capabilities of the Monitor program. Chapter 12 of this manual contains additional information on the 80386 microprocessor from a programmer's standpoint.

The Monitor Program and Programming with Interrupts

Autoboot

During the power-up sequence, the Monitor program performs a number of tests to make sure the computer is ready to function. This includes a self-test of all major circuits in the system. In addition, the Monitor program initializes all circuits and synchronizes the disk drive heads with the rest of the system. If the system detects a malfunction, one or more messages will appear on the system's display to alert the operator of a problem. Chapter 19 contains descriptions of these messages. Refer to this chapter for any problems you experience with the computer.

When all power-up tests are complete, the computer will attempt to boot the operating system in one of two ways. If the computer is equipped with a hard disk drive, it will attempt to load sector 1 of track 0 into memory and execute it. If an operating system has been installed on the drive, the computer will boot automatically. Users often refer to this process as autoboot.

If the computer only has floppy disk drives, it will attempt to load sector 1 of track 0 from the disk in drive A into memory. If no disk is in drive A, you have approximately 10 seconds to load a bootable disk into the drive. If the disk in drive A is a bootable operating system disk, then the computer will boot.

If the autoboot operation is unsuccessful in either case, an error message will be displayed. Refer to Chapter 19 for an explanation of the error message. The MS-DOS documentation describes how to prepare a floppy or hard disk system to boot MS-DOS.

There are two ways to gain access to the Monitor program. To defeat autoboot, press the ESC key after the system is first turned on. The system will display the Monitor prompt (->). To reach the Monitor program after the operating system has been loaded, press CTRL-ALT-INS. The Monitor program opening message, similar to the following, will be displayed.

MFM-300 Monitor, version 1.9

Memory Size: 640K

Press "?" for help.

->

The Monitor Program and Programming with Interrupts

NOTE: The version number will vary from the one printed in this example. Also, most desktop computers will contain 640K of system RAM; those that do not will display the actual amount of memory present.

A third method is available to access the Monitor program. This method allows you to access the Monitor program, perform operations within the Monitor program, and return to the operating system without rebooting the computer. In order to access the Monitor program in this manner, press the CTRL-ALT-RETURN keys. The Monitor prompt will be displayed in the normal fashion. When you are ready to return to the operating system, press the G key followed by the RETURN key. You may now proceed with normal operating system functions.

NOTE: Some programs do not follow MS-DOS conventions during execution. Using this method to return to the Monitor program may leave certain system registers in unexpected or undefined states. This could result in strange or unexpected results when you attempt to return to the operating system. Always make sure that important material is properly saved prior to implementing this command.

The Monitor program has six basic features. These features include: automatic power-up self-tests, user-executed tests, video commands, disk boot routines, the machine language debugger, and the basic input/output system. The rest of this chapter covers the last four topics. Chapter 19 contains a discussion of the error messages generated by the self-tests and the user-executed tests.

The Monitor Program and Programming with Interrupts

User Commands

When the Monitor prompt (->) appears on the display, you may enter commands for the Monitor program. To display the Monitor command summary, enter a question mark and press the RETURN key. A display similar to the one shown in Figure 6-1 will appear. A discussion of each command option follows Figure 6-1.

- MFM-300 Command Summary -

CMD:	Explanation	Syntax
----	-----	-----
?	Help	?
B	Boot from disk	B {[F W]}{0 1 2 3} [<partition>]
C	Color Bar	C
D	Display memory	D [<range>]
E	Examine memory	E <addr>
F	Fill memory	F <range>,<byte> "<string>"...
G	Execute (Go)	G [= <addr>] [,<breakpoint>]...
H	Hex math	H <number1>,<number2>
I	Input from port	I <port>
M	Move memory block	M <range>,<dest>
O	Output to port	O <port>,<value>
R	Examine Registers	R [<register>]
S	Search memory	S <range>,<byte> "<string>"...
T	Trace program	T [<count>]
U	Unassemble program	U [<range>]
V	Set Video/Scroll	V [M<mode>] [S<scroll>] [100 300]
	Where <range> is:	<addr>{, <addr> L<length>}
TEST	Extended diagnostics	TEST
SETUP	Define Hardware Setup	SETUP

Copyright (C) 1987. by Zenith Data Systems

Figure 6-1. MFM-300 Command Summary Menu

The Monitor Program and Programming with Interrupts

Video Commands and Disk Boot Routines

The Monitor program contains three video-related commands and one general-purpose disk boot command. These are summarized in Table 6-1 and described in the following paragraphs.

Table 6-1. Video and Disk Commands

COMMAND	DESCRIPTION	SYNTAX
?	Help	?
C	Display color bar	C
V	Set video/scroll mode V	[M<mode>] [S<scroll>][100][300]
B	Boot disk	B [F W][0 1 2 3][:Partition]

Help

Syntax: ?

Example: ? RETURN

Use the help command to display a summary of the Monitor program commands. The summary contains a list of each Monitor program command, followed by its title and the syntax for entering the command. If you enter the example as shown, the system will display the command summary illustrated in Figure 6-1.

Display Color Bar

Syntax: C

Example: C RETURN

The color bar command will display 16 color bars on a color display or three bars on a monochrome display. You can use this display to adjust a color or monochrome CRT monitor. If you enter the example, the screen will clear and display the color bar pattern.

The Monitor Program and Programming with Interrupts

Set Video/Scroll Mode

Syntax: V [M<mode>] [S<scroll>][100][300]

Examples: V M3 RETURN
 V S0 RETURN
 V M6 S2 RETURN

Use the set video/scroll mode command to select one of eight video modes (this computer implements seven modes) and one of three scrolling modes as described in Table 6-2. The first example sets the display to video mode 3 (color, 80 characters by 25 rows). The second example sets the scrolling mode to 0 (software-controlled scrolling). The third example sets the video mode to 6 (monochrome graphics 640 × 200 resolution) and scrolling mode 2 (hardware-controlled smooth scrolling).

Table 6-2. Video and Scroll Modes

MODE	DESCRIPTION
M0	40 characters by 25 rows text, monochrome display on the RGB output.
M1	40 characters by 25 rows of text, color display on the RGB output.
M2	80 characters by 25 rows text, monochrome display on the RGB output. Individual video pages may be scrolled without affecting other video pages.
M3	80 characters by 25 rows text, color display on the RGB output or as a gray-scale display on a monochrome display. Individual text pages may be scrolled.
M4	320 × 200 pixel resolution graphics, color display on the RGB output or as a gray-scale display on a monochrome display. The text display is 40 characters by 25 rows.
M5	320 × 200 pixel resolution graphics, monochrome display on the RGB output. Text is displayed at 40 characters by 25 rows.
M6	640 × 200 pixel resolution graphics, monochrome display on RGB output. All three scrolling modes are available. The text display is 80 characters by 25 rows.
M7	80 characters by 25 rows, monochrome text display. This mode requires a TTL-compatible monochrome graphics adapter card.
S0	Software scrolling. This mode operates in all video modes and is the most common mode used in PC-compatible software. When material is scrolled, the display moves a line at a time.

The Monitor Program and Programming with Interrupts

Table 6-2 (continued). Video and Scroll Modes

MODE	DESCRIPTION
S1	Hardware jump scrolling. This mode is available for video modes 3 through 6 and is similar in action to software scrolling. In this mode the hardware controls the scrolling action, rather than the software.
S2	Hardware smooth scrolling. This mode works only in video mode 6. When the material is scrolled, the display moves a partial line at a time providing a much smoother appearance to the scrolling action.

Boot Disk

Syntax: B [F | W][0 | 1 | 2 | 3][:partition]

Examples: B RETURN
 BF1 RETURN
 BW:1 RETURN

This command manually boots the operating system from a bootable disk installed in any disk drive attached to the computer. You may specify floppy or hard drives and the specific drive you wish to boot from.

In the first example, the computer will boot the default drive. When the computer first powers up, the default drive will either be the hard disk drive or floppy disk drive A, depending on the configuration of your system. If you manually boot from another disk drive, such as drive B, that drive will become the new default drive. The default setting will remain the same until you boot from a different drive or turn the system off. If the system is turned off, the default drive once again becomes the boot drive.

In the second example, the computer will boot from drive B if it is present. The F in the syntax is optional and specifies 3.5-inch or 5.25-inch floppy disk drives. The number 1 in the example would specify floppy drive unit 1, also referred to as drive B.

The Monitor Program and Programming with Interrupts

In the third example, the computer will boot from the second partition on the hard disk drive. The letter W is optional and specifies boot operations from a hard disk drive.

Chapter 19 describes several different error messages that the computer can generate during the boot process.

The Machine Language Debugger

The Monitor program contains a set of machine language debugging commands, listed in Table 6-3. The syntax in this table is clearer than that shown in the command summary in Figure 6-1. Examples and descriptions follow the table.

The message ^ Invalid Command! is the Monitor program's syntax error message. The caret points to the exact spot where the first syntax error occurs. The system does not recognize additional syntax errors after the first error is detected. This error does not lock up the system. Therefore, the command can be reentered or another command entered in its place.

Table 6-3. Machine Language Debugger Commands

COMMAND	DESCRIPTION	SYNTAX
D	Display memory	D <address>[L<length> <offset>]
E	Examine memory	E <address>
F	Fill memory	F <start address>, <range>, <data list>
G	Go (Execute)	G [= <address>][, <breakpoint>]...
H	Hex math	H <number1>, <number2>
I	Input from port	I <port address>
M	Move memory block	M <start address>, <range>, <new address>
O	Output to port	O <port address>, <value>
R	Examine Registers	R [<register>]
S	Search memory	S <start address>, <range>, <data list>
T	Trace program	T [= <address>][, <value>]
U	Unassemble program	U [<address>][<length> , <offset>]

The Monitor Program and Programming with Interrupts

Display Memory

Syntax: D <address>[L<length> | <offset>]

Examples: D 1000:0 F RETURN
 D DS:7000 L200 RETURN

Use this command to display the contents of the specified portion of memory. If the length is not specified in the command line, then the debugger will display 128 bytes of data, starting at the specified address. The resulting display contains three sections. The left-most column is the base address and offset of the first byte in the second part of the display. The second part uses hexadecimal format to display 16 bytes of data. The right-most part of the display contains the ASCII characters represented by the data in the second part of the display. If any byte is outside the ASCII character range, a period will be printed in its place. The amount of data you choose to display can be controlled using either the optional length specification or the offset.

If the first example is entered, a display of 16 bytes of memory will result, starting at address 1000:0. If that portion of memory is empty (containing only nulls, or 0s), the resulting display will be:

```
1000:0000 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 .....
```

If the second example is entered, the resulting display will contain 200H bytes of memory. This display will start at the address offset located 7000H bytes from the base address of the data segment. The base address is stored in CPU segment register DS.

If the memory range is greater than can be displayed on the screen, the data will scroll. Use CTRL-S to stop and CTRL-Q to continue the screen action.

The Monitor Program and Programming with Interrupts

Examine Memory

Syntax: E <address>

Example: E 1000:100 RETURN

Use the examine command to examine or change memory one byte at a time. When this command executes, the display will show the byte at the specified memory address. The command will wait for an entry from the keyboard. Enter a hexadecimal value from 0 to FF to modify the current memory location. Press the space bar to examine the next byte of memory or the minus (hyphen) key to display the previous byte of memory. Press the RETURN key to complete this command and return to the Monitor program prompt.

CAUTION

You can examine and modify any addressable memory location, including those reserved for interrupts and interrupt vectors. Therefore, be careful that you do not accidentally modify routines that the Monitor program uses or you could lock up the computer. If this occurs, turn the computer off to reset it.

Fill Memory

Syntax: F <start address>,<range>,<data list>

Example: F 1800:0,1FF,"TEST",20,54,45,53,54,20 RETURN

Use the fill command to enter one or more bytes of data directly into memory. You can use ASCII characters, delineated by quotation marks, or hexadecimal format. Each byte and ASCII text word requires a comma for separation. Data in the list will be reused as often as necessary to fill the specified memory range. You must specify the starting address, range, and data before the command will function. In the example, 1FFH bytes, starting at memory address 1800:0, will fill with repeating strings of the data. The entire string is used, which includes the material in quotes plus the material expressed as hexadecimal values. The result of this example is the string TEST TEST.

The Monitor Program and Programming with Interrupts

CAUTION

You can use the fill command to modify any addressable memory range, including those reserved for interrupts and interrupt vectors. Therefore, be careful that you do not accidentally modify routines that the Monitor program uses or you could lock up the computer. If this occurs, turn the computer off to reset it.

Go (Execute)

Syntax: G[= <address>][, <breakpoint>]...

Examples: G 1000:0 RETURN
 G 100 RETURN
 G=1000 100 RETURN

You can use the go command to transfer control from the Monitor program to a machine language or user program. The transfer will occur at the address specified in the command line or in the instruction pointer and code segment of the CPU. Also, as a debugging aid, you may specify breakpoints (points in the program where you want the program to halt) in the command line.

In the first example, control will transfer to the address specified in the CPU registers and a breakpoint will be set at memory address location 1000:0. In the second example, the Monitor program will transfer control to the address specified in the CPU registers and set a breakpoint at the memory address offset. This address is located 100H bytes from the location of the first byte of code to be executed. In the third example, the Monitor program sets the code segment register to 1000. Control is then transferred to this address and the Monitor program sets a breakpoint at memory offset 100H.

When the go routine encounters a breakpoint, the Monitor program saves the status and register information. This information is displayed as a register dump (refer to the examine/modify registers command). The screen will display the status of the CPU's registers and flags as well as the current instruction executed just prior to the breakpoint.

The Monitor Program and Programming with Interrupts

You may set up to eight breakpoints and the debugger will halt when it encounters one of them. Once a breakpoint is found, the debugger will not remember the other breakpoints, since they are stored as parameters to the command line.

NOTE: You must set breakpoints to addresses that contain valid instruction codes, otherwise the computer will lock up.

Hex Math

Syntax: H <number1>,<number2>

Example: H 2a,28 RETURN

Use the hex math command to compute the sum and difference of two hexadecimal numbers. The debugger always subtracts the second number from the first. In the example, the results will display as Sum: 0052 Diff: 0002.

Input From Port

Syntax: I <port address>

Example: I 3F8 RETURN

Use the input command to obtain a byte of data from a port address. In the example, a single byte from port 3F8H will be displayed on the screen. You can address most programmable devices used in this computer through port addresses.

Move Memory Block

Syntax: M <start address>,<range>,<new address>

Example: M 0:1000,107F,0:2000 RETURN

The Monitor Program and Programming with Interrupts

Use the move command to copy a specified block of memory to another specified location in memory. In the example, the Monitor program will copy a block of memory 107FH bytes long, starting at memory address 0:1000, to a memory location that starts at 0:2000.

CAUTION

You can use the move command to move a block of memory from any addressable memory location to any other addressable memory location, including one reserved for interrupts and interrupt vectors. Therefore, be careful that you do not accidentally modify memory that contains routines used by the Monitor program or you could lock up the computer. If this occurs, turn the computer off to reset it.

Output To Port

Syntax: O <port address>,<value>

Example: O 2F8,00 RETURN

Use the output command to send a byte of data out a port address. In the example, the Monitor program will send the value of 0H out port address 2F8H. The computer can address most programmable devices through port addresses.

Examine/Modify Registers

Syntax: R [<register>]

Examples: R RETURN
 R CS RETURN
 R FL RETURN

Use the examine registers command to examine and alter the contents of the CPU's registers and flags. In the first example, the computer performs a register dump. The dump will display the current instruction, along with the contents of all registers and flags.

The Monitor Program and Programming with Interrupts

In the second example, the display will contain the contents of the CS register and the computer will wait for an entry from the keyboard. Enter a hexadecimal value followed by the RETURN key, to modify the contents of the register. If the value you enter is not valid, the register's contents will not change. Press the RETURN key by itself to leave the contents of the register unchanged.

In the third example, the display will contain the status flags. Table 6-4 contains the two-letter abbreviation that describes the status of each flag. The computer will wait for an entry from the keyboard. Enter the two-letter abbreviation to set the status of a flag, followed either by a space and another two-letter abbreviation for another flag or by the RETURN key to execute the command. If you make more than one entry for the same flag, the last value entered will be the one that remains. Press the RETURN key by itself to leave the flags unmodified.

Table 6-4. Processor Status Flag Codes

FLAG	ON	OFF
Auxiliary carry	AC(auxiliary carry)	NA(no auxiliary carry)
Carry	CY(carry)	NC(no carry)
Direction	DN(down)	UP(up)
Interrupts	EI(enabled)	DI(disabled)
Overflow	OV(overflow)	NV(no overflow)
Parity	PE(even)	PO(odd)
Sign	NG(negative)	PL(plus)
Zero	ZR(zero)	NZ(not zero)

NOTE: This table does not represent all flags available within the 80386 microprocessor. Refer to Chapter 12 for additional information.

The Monitor Program and Programming with Interrupts

Search Memory

Syntax: **S** <start address>,<range>,<data list>

Example: **S** 0:2000,1000,"TEST",20,"TEST" RETURN

Use the search command to search the specified area of memory for the occurrence of specific data strings or bytes of information. In the example, the search routine will examine 1000H bytes of memory, starting at memory location 0:2000, for the occurrence of TEST TEST. Notice that quotation marks enclose the ASCII text; otherwise, you must enter data in hexadecimal format. You must separate each item in the data list from the other items in the list with commas. If you do not provide any data in the list, the Monitor program will return the ^ Invalid Command! error message.

Trace User Program

Syntax: **T** [= <address>][,<value>]

Examples: **T=400:0,200** RETURN
 T 5 RETURN

Use the trace command to single-step the execution of a program. Each time you execute the command, it executes the next instruction and performs a register dump. The dump will display the current instruction and the contents of the registers and flags. Enter T by itself to execute the instruction pointed to by the current code segment and IP register. Use the examine/modify registers command to change these values and to specify a new memory location for trace operation.

In the first example, the routine places the value following the equal sign (=) in the CS and IP registers. Trace will use this value as the starting address and will execute 200H times. Since this is a large number, the register dumps will fill and scroll the screen. Use CTRL-S to stop and CTRL-Q to start the scrolling action.

The Monitor Program and Programming with Interrupts

In the second example, trace will start at the current address stored in the CS and IP registers of the CPU. The command will repeat five times.

CAUTION

The trace command does not contain extensive error trapping routines to prevent accidental operation of the computer. It is possible that timing-dependent operations, such as reading or writing to a disk, could cause problems and, in the case of disk input/output operations, damage the media itself. Therefore, do not use trace to step through the Monitor program or interrupt service routines.

Unassemble

Syntax: U [<address>[<length> | ,<offset>]

Examples: U RETURN
 U 1000 RETURN
 U 0:1000 RETURN

Use the unassemble command to disassemble a section of memory into assembly language mnemonic format. The resulting display resembles an assembly language source code listing, but contains no comments. The Monitor program will treat data areas that contain ASCII text as code. Therefore, it is possible to create meaningless assembly language code from raw text or other data.

In the first example, the routine will begin disassembly in the current code segment (pointed to by the CS register) at the address pointed to by the IP register. The routine will disassemble and display 32 bytes (20H) of data. The number of lines displayed on the screen will vary depending on the number of bytes generated by each instruction.

In the second example, disassembly starts at the memory location offset 1000H bytes from the value of the CS register. In the third example, disassembly starts at memory location 0:1000. In all three cases, the routine will disassemble and display 32 bytes of data.

The Monitor Program and Programming with Interrupts

NOTE: Although unassemble modifies the values in the CS and IP registers, the routine stores the original values in a stack. If you use the examine/modify registers command, the routine will restore the original values and subsequent use of the unassemble command will use these values.

Although not shown in an example, you can use the length parameter to disassemble more or less than 32 bytes of memory. Since this may result in a display that scrolls the screen, use CTRL-S to stop and CTRL-Q to start the scrolling action.

Programming with Interrupts

The basic input/output system is the portion of the Monitor program that establishes the machine-level input/output routines. These routines provide control at the device level of the main input/output devices for the computer. Use software interrupts to call a routine and perform a function.

To use an interrupt, place a specific code in register AH of the CPU and execute an INT instruction. As a rule, the Monitor program routines will preserve the values in all CPU registers except for AX and the flags. Modifications of other registers may occur upon completion of the current task only if they are returning a value to the program that executed the function.

The firmware generates software interrupts when certain events occur within the computer. You can use these events in your programs by patching the events' interrupt vectors for your routines. If you do this and run your routines with other programs that use or modify the same interrupts or memory locations, you may experience unexpected results.

Hardware Interrupts

Although this part of the technical manual is concerned with the software interrupts, you need to understand the difference between interrupts generated by application software or the firmware of the computer and interrupt requests made by the hardware.

The Monitor Program and Programming with Interrupts

The 8259 programmable interrupt controller manages hardware interrupts. Two 8259s in cascade provide a total of 16 interrupt levels in this machine. In the event that more than one request for an interrupt occurs at the same time, the controller will manage these requests on a priority basis.

The highest priority is assigned to IRQ0 (interrupt request 0) and the lowest to IRQ15. By sending a bit mask to port 021H, the programmer can control whether a particular interrupt can be processed when requested. For more information on the 8259, refer to Chapter 14.

Table 6-5 describes the usual hardware interrupt requests and the software interrupts they trigger. The normal order of precedence is used in the table. Jumpers or other hardware factors may change the source of the interrupt request in some computer installations, but this is not recommended.

Table 6-5. Hardware Generated Interrupt Requests

HARDWARE INTERRUPT REQUEST	SOFTWARE INTERRUPT	USUAL SOURCE OF INTERRUPT
IRQ0	08H	Time-of-day timer
IRQ1	09H	Key pressed
IRQ2	0AH	Slave controller interrupt
IRQ3	0BH	Communications (COM2) or PC bus
IRQ4	0CH	Communications (COM1) or PC bus
IRQ5	0DH	Parallel printer (LPT2) or PC bus
IRQ6	0EH	PC bus
IRQ7	0FH	Parallel printer (LPT1) or PC bus
IRQ8	—	Real-Time Clock
IRQ9	—	IRQ2 on PC bus
IRQ10	—	AT bus
IRQ11	—	AT bus
IRQ12	—	AT bus
IRQ13	—	Coprocessor interrupt (80287 or 80387)
IRQ14	—	AT bus
IRQ15	—	AT bus

The Monitor Program and Programming with Interrupts

Using a Software Interrupt

Some of the interrupts discussed in this manual are initialized by the operating system. Since MS-DOS is the standard operating system for these computers and helps maintain PC-compatibility, those interrupts that are considered universal (not specific to a particular version of MS-DOS) are included in the summary presented later in this chapter. Those based on MS-DOS that are subject to change between versions are discussed in the *Programmer's Utility Pack* for MS-DOS. Other operating systems and MS-DOS for other computers (not PC-compatible) may or may not initialize the same or similar interrupts.

NOTE: Any software interrupt that has not been initialized will jump to a return (RET) function (no operation).

When you prepare to use or modify an interrupt routine, there are a couple of factors you should keep in mind:

- All parameters that are passed to and from interrupt routines must go through the CPU's registers. Where more than one function may be performed by a routine, or where additional parameters are required by a routine, more than one register will be used by the interrupt.
- Most of the interrupt routines will preserve the values in the CPU registers except when a value will be returned to a specific register. When you write or modify routines for these interrupts, you should plan on preserving, where possible, the values of the CPU registers.

Each interrupt routine is pointed to by a dedicated interrupt vector (address). To execute one of the interrupt routines, you must first load the specified parameters into the registers of the CPU. Then perform an INT xxH instruction where xxH is the interrupt number. For instance, the code example in Listing 6-1 executes the INT 11H interrupt and then tests for and returns the number of drives connected to the system as established by the configuration switches.

The Monitor Program and Programming with Interrupts

Since no parameters were required by INT 11H, none are loaded. Note however, that the results are returned in register AL, where they are tested by the routine.

Listing 6-1. Sample Code Segment

```

INT 11H          ;test for number of drives
PUSH AL          ;retrieve status byte
AND AL,1B        ;temporarily store results
TEST AL,0        ;isolate bit 0
JZ ZERO_DRIVES   ;test for drives present
POP AL           ;if no drives, jump to routine
AND AL,1100000B  ;pop results back off stack
MOV CL,5         ;isolate bits 5 and 6
SHR AL,CL        ;set for shift of 5 bit-places
INC AL           ;shift data right 5 bits
                 ;increment results by 1 so
                 ;that true number of drives
                 ;is now in register AL

```

Modifying an Interrupt

To modify an interrupt, you must patch the interrupt vector with the address of your routine as outlined in the following procedure (MS-DOS environment). For more information, refer to the *Programmer's Utility Pack* for your version of MS-DOS.

1. Use the MS-DOS function request Get Interrupt Vector (35H) to read the current value of the interrupt vector. Save this value in your code segment.
2. Use the MS-DOS function request Set Interrupt Vector (25H) to set the address of your interrupt routine.
3. Do not execute an IRET instruction at the end of your routine. Instead, execute a JMP DWORD PTR to the address saved in step 1. This allows multiple background tasks to have a chance at the interrupt. Note that the final routine executed during an interrupt sequence is an IRET, but only after the CPU registers have been restored.

The Monitor Program and Programming with Interrupts

4. Do not modify any CPU registers unless specifically stated in the discussion of that interrupt. Upon entry to an interrupt routine, the ES, DS, BX, CX, DX, AX, BP, SI, and DI registers contain information that needs to be saved. Therefore, it is a good idea to push the information onto the stack and then restore it following the execution of your routine.

Software Interrupt Summary

Table 6-6 lists the interrupts, their functions, the initializing system (the Monitor program MFM-300, MS-DOS, or application software), and the chapter where you can find a complete discussion. Interrupts that show dashes (—) in each column are either reserved for future use or are not generally used by PC-compatible computers; more specifically, they are not implemented in Zenith Data Systems computers or software.

Table 6-6. Interrupt Summary

INTERRUPT	INITIALIZED BY	CHAPTER	FUNCTION
00H	DOS	7	Divide by zero
01H	DOS	7	Single step
02H	MFM-300	7	Nonmaskable interrupt
03H	DOS	7	Software breakpoint
04H	DOS	7	Arithmetic overflow
05H	MFM-300	9	Print screen
06H	—	—	
07H	—	—	
08H	MFM-300	7	Timer (time-of-day)
09H	MFM-300	8	Key pressed
0AH	MFM-300	7	Real-time clock
0BH	Software	9	Communications (COM2)
0CH	Software	9	Communications (COM1)
0DH	Software	9	Alternate parallel printer (LPT2)
0EH	MFM-300	10	Floppy disk drive
0FH	Software	9	Parallel printer (LPT1)
10H	MFM-300	11	Video input/output
11H	MFM-300	7	Equipment configuration
12H	MFM-300	7	Memory size
13H	MFM-300	10	Disk input/output
14H	MFM-300	9	Serial input/output

The Monitor Program and Programming with Interrupts

Table 6-6 (continued). Interrupt Summary

INTERRUPT	INITIALIZED BY	CHAPTER	FUNCTION
15H	—	—	
16H	MFM-300	8	Keyboard input/output
17H	MFM-300	9	Printer input/output
30H	MFM-300	9	Parallel/serial configuration
19H	MFM-300	10	Bootting an operating system
1AH	MFM-300	7	Set/read the time of day
1BH	MFM-300	8	Keyboard break
1CH	MFM-300	7	Tick timer
1DH	MFM-300	11	Video initialization
1EH	MFM-300	10	Disk parameters
1FH	MFM-300	11	Defining characters

System Organization

Table 6-7 provides an overall map of the system's addressable memory. Table 6-8 is a general map of the system's ports. The information presented in these tables is subject to change as new products are introduced for this family of computers. The values represented by question marks vary, according to the version of MS-DOS and the user-installed .COM and .EXE files.

NOTE: This computer does not support a light pen port. Some computer games use this feature and may not operate correctly.

Table 6-7. System Memory Map

ADDRESS RANGE	DESCRIPTION
00000H–9FFFFH	System memory. This area is further divided by the Monitor program and DOS as follows:
(00000H–003FFH)	Interrupt vector table (addresses)
(00400H–0047FH)	Monitor program compatible data segment
(00500H–?????H)	IO.SYS (part of DOS)*
(?????H–?????H)	DOS.SYS (part of DOS)*
(?????H–?????H)	Resident portion of COMMAND.COM (part of DOS)
(?????H–?????H)	User-installed .COM and .EXE files

The Monitor Program and Programming with Interrupts

Table 6-7 (continued). System Memory Map

ADDRESS RANGE	DESCRIPTION
(?????H-?????H)	Transient portion of COMMAND.COM
(?????H-9FFFFH)	Open for general use
A0000H-BFFFFH	Video RAM area
C0000H-C7FFFH	EGA BIOS ROM area
C8000H-C9FFFH	Hard disk BIOS ROM area
CA000H-CFFFFH	Open area
D0000H-DFFFFH	EMS window area
E0000H-EEFFFH	Optional ROM area
F0000H-F07FFFH	Scratchpad RAM
F8000H-FFFFFFH	System ROM 1 (Monitor program)
I00000H-FDFFFFH	Open area (with optional expansion memory)
FE0000H-FFFFFFH	System Monitor ROM

*IO.SYS and DOS.SYS are representative of the names of hidden files that are part of MS-DOS. The actual names of these files can vary from version to version.

Table 6-8. System Port Map

PORT ADDRESSES	DESCRIPTION
000H-01FH	8-bit DMA processor
020H-03FH	Interrupt controller #1 (8259)
040H-05FH	System timer (8254)
060H-06EH	System control processor
061H-06FH	Port B
070H-07EH	NMI enable
080H-09FH	DMA page registers (gate array 1)
0A0H-0BFH	Interrupt controller #2 (8259)
0C0H-0DFH	16-bit DMA controller
0EEH	Fast GATE A-20
0EFH	Fast reset CPU
0F0H	Clear coprocessor busy latch
0F1H	80287 reset
0F2H	Diagnostic LED port
0F4H, 0F5H	Slow mode control
0F9H, 0FBH	Scratchpad RAM enable
200H-20FH	Game input/output port (not implemented)
258H	EMS Address
278H-27FH	Parallel port 2 (LPT2)

The Monitor Program and Programming with Interrupts

Table 6-8 (continued). System Port Map

PORT ADDRESSES	DESCRIPTION
2F8H-2FFH	RS-232C serial input/output interface #2 (COM2)
320H-323H	Hard disk drive
378H-37FH	Parallel printer #1 (LPT1)
3F0H-3F7H	Floppy disk controller
378H-37FH	Parallel port 1 (LPT1)
3F8H-3FFH	RS-232C serial input/output interface #1 (COM1)

Monitor Program Jump Vectors

The Monitor program sets up two jump vectors that can be used by applications programs. They are located at F000:FFF0, which is the power-up reset vector, and F000:FFED, which is an unconditional jump to the Monitor program (the Monitor program prompt will appear on the screen).

Chapter 7

System and CPU Interrupts

This chapter describes the system and CPU interrupts, programming sound, and addressing user memory.

Programming System and CPU Interrupts

Table 7-1 defines the general system interrupts used in this computer. For information on the use and programming of the software interrupts, refer to Chapter 6.

Table 7-1. System and CPU Interrupts

INTERRUPT	OPERATION
00H	Divide by zero
01H	Single step
02H	Nonmaskable interrupt
03H	Software breakpoint
04H	Arithmetic overflow
08H	Timer (time-of-day)
0AH	Real-time clock
11H	Equipment configuration
12H	Memory size
15H	Device control
1AH	Set/read the time of day
1CH	Tick timer
4AH	User alarm interrupt
76H	RTC alarm

Divide by Zero (INT 00H)

INT 00H is the divide by zero interrupt. This interrupt occurs if a divide instruction produces a quotient too large to fit in the result register (such as dividing a value by 0). The operating system initializes this routine. The routine will display *Divide Overflow* and return control to the operating system.

System and CPU Interrupts

You must set up a vector to intercept DIV and IDIV instructions if you do not want control returned to the operating system. That way, you can test for the error condition and prevent program control from returning to DOS. For instance, BASIC uses this method to retain control when a divide by zero condition occurs while executing BASIC functions.

Single Step (INT 01H)

INT 01H is the single step interrupt used for executing a single machine instruction at a time. The CPU calls this interrupt when an instruction executes with the trace flag (TF) set.

MS-DOS and the DEBUG command commonly use this interrupt, as does the Monitor program's trace command. The routine receives initialization instructions from the calling program (issued by DOS, the Monitor program, etc.). Because of this, you must also initialize the routines you want executed when you call this interrupt.

Non-maskable Interrupt (INT 02H)

INT 02H is the nonmaskable interrupt. Hardware external to the 80386 initializes this interrupt. Most Zenith Data Systems computers use this interrupt to indicate when a power-down condition has started. Also, the 80287/80387 numeric data coprocessor uses this interrupt in its normal operation.

Normally, this interrupt is enabled. One exception however, is during the power-up sequence while the self-tests are being run. If you need to disable this interrupt (turn it off), send 00H to port A0H. To enable the NMI, send 80H to the A0H.

Software Breakpoint (INT 03H)

INT 03H is the software breakpoint interrupt. When the processor encounters a breakpoint in a program, it will execute an INT 03H instruction, calling the interrupt routine.

System and CPU Interrupts

The Monitor program's debugging routines and the MS-DOS DEBUG command allow you to set breakpoints in machine language code. Whenever the Monitor program or MS-DOS DEBUG encounter a breakpoint, control returns to the last command level.

Arithmetic Overflow (INT 04H)

INT 04H is the arithmetic overflow interrupt. The INTO instruction (interrupt on overflow) will generate this interrupt whenever the overflow flag (OF) is set. Arithmetic and logic operations set the overflow flag.

Timer (Time-of-Day) (INT 08H)

INT 08H is the timer interrupt. The output of counter 1 of the 8254 programmable interval timer initiates this interrupt 18.2159 times per second, or every .054897095 seconds. The CPU clock speed does not affect the operation of this timer. The timer performs such functions as keeping track of the time, timing out disk motors, and calling the timer tick interrupt (1CH).

In many computers, the timer keeps track of the time-of-day in a 32-bit word (sometimes called a double-word). When the count reaches approximately 1803D8H (1,573,848 decimal), a flag will be set to 1. This flag indicates that the timer has rolled past midnight into a new day. The value of this word is read using interrupt 1AH. The hardware establishes this interrupt as IRQ0.

In this computer, a real-time clock maintains the time of day. The real-time clock IC is described in Chapter 14.

Real-Time Clock (INT 0AH)

INT 0AH is the real-time clock alarm interrupt. The real-time clock starts when a hardware interrupt request (IRQ3) or when the alarm from the real-time clock IC takes place. Function 83H (event wait) and function 86H (wait) of INT 15 set and handle the alarm. The hardware interrupt request takes place approximately 1,024 times a second.

System and CPU Interrupts

Most systems do not use this interrupt. The hardware within the computer essentially controls this interrupt. This makes the interrupt unique to individual machines such as this computer. Because of this fact, it should not be called by user programs.

Equipment Configuration (INT 11H)

INT 11H is the request equipment configuration interrupt. This interrupt reports the configuration of the equipment in a 16-bit word (register AX). Since this interrupt is common to PC-compatible equipment, you need to be aware of all possible responses and what they mean. Refer to Tables 7-2, 7-3, and 7-4 for a description of each bit in the data returned in register AX.

Table 7-2. Register AX Report from INT 11H

BIT	DESCRIPTION
0	If set (1), floppy disk drives are present in the system; if clear (0), no floppy disk drives are installed.
1	If set (1), the coprocessor is present; if clear (0), the coprocessor is not installed.
2-3	These two bits indicate the device size of the RAM chips installed in the computer. Bits 2 and 3 are always set (1), indicating that either 64- or 256-kilobit devices are installed. Some early PC-type computers used 16- and 32-kilobit devices in memory.
4-5	These two bits indicate the initial video mode at powerup. Refer to Table 7-3.
6-7	These two bits report the number of floppy disk drives installed in the computer according to the hardware switch or firmware settings. Refer to Table 7-4.
8	Unused in Zenith Data Systems computers.
9-11	The value in these three bits indicate the number of RS-232 ports in the system. The standard input/output on this system emulates the number of IBM PC input/output ports, so the value is one.
12	If set (1), a game card is present; it is clear (0) in this computer.
13	Not used.
14-15	The value in these two bits is the number of printers installed.

System and CPU Interrupts

Table 7-3. Bits 4 and 5: Video Initialization

BIT 4	BIT 5	DESCRIPTION
0	0	80 × 25 text mode on an EGA card
0	1	40 × 25 text mode on a color card
1	0	80 × 25 text mode on a color card
1	1	80 × 25 text mode on a monochrome card

Table 7-4. Bits 6 and 7: Disk Drive Count

BIT 6	BIT 7	DRIVE COUNT
0	0	1 drive
0	1	2 drives
1	0	3 drives
1	1	4 drives

Memory Size (INT 12H)

INT 12H is the memory size interrupt. It returns the number of contiguous 1K blocks of user memory in the system in register AX. For example, if the value in AX is 256 following this interrupt, then the computer has determined that 256K of memory is available in the system.

Device Control (INT 15H)

The device control interrupt (INT 15H) functions as the entry point for device control and system request functions. This interrupt normally supports cassette I/O functions in PC-compatible computers, but not in this system. Instead, a number of other functions are available using this interrupt. Table 7-5 lists the normal functions available to INT 15H. Table 7-9 lists the extended functions that are reserved for this system only.

The device control interrupt functions are activated in the normal fashion. Place the function value in the AH register and call interrupt 15H.

System and CPU Interrupts

Table 7-5. INT 15 Functions

FUNCTION	DESCRIPTION
00-03H	Cassette I/O (not supported)
04-7FH	User-defined functions
80H	Open device
81H	Close device
82H	Terminate program
83H	Set non-busy wait alarm
84H	Read joystick
85H	SYSTEM REQUEST key pressed
86H	Busy wait
87H	Block move request
88H	Size extended memory
89H	Set processor in virtual mode
90H	Device busy loop entry point
91H	Set interrupt complete entry point

NOTE: If the user calls a function in the range of 04H through 7FH without creating a function routine, the value 86H will be returned in the AH register and the CY flag bit will be set.

Function 80H: Open Device — This user-defined function is not implemented in this computer system. This function is provided to support multitasking operating systems. Two input parameters are required when using this function. The BX register must contain the device ID and the CX register contains the process ID.

Function 81H: Close Device — This user-defined function is not implemented in this computer system. This function is provided to support multitasking operating systems. Two input parameters are required when using this function. The BX register must contain the device ID and the CX register contains the process ID.

Function 82H: Program Terminate — This user-defined function is not implemented in this computer system. This function is provided to support multitasking operating systems. When terminating a program in a multitasking environment, the BX register contains the device ID.

System and CPU Interrupts

Function 83H: Set Non-Busy Wait Interval — This function provides a mean of internally monitoring elapsed time intervals. The CX and DX registers contain the wait interval count in milliseconds. The CX register contains the most-significant word and the DX register contains the least-significant word. When the specified time period has passed the ROM will set the high order bit of a byte specified in memory. The ES register contains the segment address of the byte and the BX register contains the offset.

Function 84H: Read Joystick — This function allows the user to obtain information concerning the current joystick position or the status of the joystick triggers. If the value in the DX register is a zero, the function will report the status of the joystick triggers. If the DX register contains a value of one, the function will return the current joystick position.

When the function returns it will place the requested information in different registers. The AL register contains the trigger status in bit positions 4–7. The function reports position information for two joysticks. The AX register contains the X coordinate of joystick A; the Y coordinate is in the BX register. The CX and DX registers contain the respective X and Y coordinates for joystick B. If no joystick is installed on the system the values for these registers are undefined.

If the value in the DX register is not correct on entry into the function, it must be either a zero or a one, the function will return the value 86H in the AH register and set the carry bit.

Function 85H: System Request Key Pressed — This function lets the user know when the SYSTEM REQUEST key has been pressed. The function will place a value in the AL register when the key is pressed and released. If the AL register contains a zero, the key is in the down position. If the AL register contains a one, the key is in the up position.

Function 86H: Busy Wait Interval — This function is very similar to Function 83H, the main difference is that the ROM will not return to the user until the specified time period has passed. In this case, the ROM essentially calls function 83H.

System and CPU Interrupts

Upon entry into the function, the CX register must contain the most-significant word of the wait interval in milliseconds. The DX register must contain the least-significant word.

The function will return one of two possible values upon completion. If the requested time interval has passed, the NC bit will be set. If another wait interval request is presently active, the CY bit will be set.

Function 87H: Block Move — This function will move a block of memory, in words, from one section of memory to any another section in the 16M address space of the computer. The function is designed to allow the user a means of moving memory data above the 1M address boundary when the processor is in protected mode. The block of memory you move can be no larger than 64K.

In order to move a block, you must place the number of words you want moved in the CX register. The ES:SI register pair form a pointer to the descriptor table used during the move. The descriptor table is a standard descriptor table like those described in Chapter 12. It is built by your routine prior to calling this function. Figure 7-1 illustrates the general format of the descriptor table. When you call the function, the ROM will place the processor in protected mode, move the block of memory, and return the processor to its original state. Table 7-6 describes the descriptor table used for block moves.

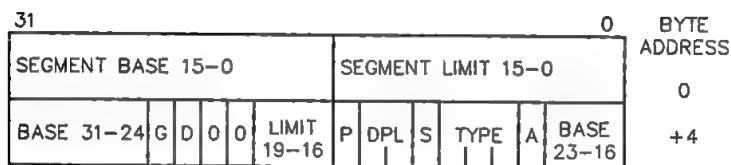


Figure 7-1. Block Move Descriptor Table

System and CPU Interrupts

Table 7-6. Block Move Table Description

DESCRIPTOR	OFFSET	INITIALIZED BY	INITIALIZED VALUE
1 Dummy	00	User	0
2 GDT	08	User	0
3 Source	10	User	Source address
4 Target	18	User	Target address
5 Code	20	User	0
6 Stack	28	User	0

If an error occurs during the block move, the ROM will provide an error code and set a flag. The AH register will contain the error code and the CY bit in the flag's register will be set. The value returned in the AH register will be one of the following values:

- AH = 0 : Block move successful
- AH = 1 : RAM parity error
- AH = 2 : Exception interrupt error
- AH = 3 : Address 20 line failure

If the block move was successful, the zero flag bit will be set.

NOTE: The system is not intelligent enough to protect itself during block moves. If you should, for example, move a block of memory into an address area occupied by important system data files, you could easily halt the computer system. This could also destroy data in the computer. Always exercise extreme care when you perform block moves.

Function 88H: Extended Memory Size Determination — This function returns the number of contiguous 1K memory blocks above the 1M boundary. When you call the function, it will return the number of blocks in the AX register.

Function 89H: Set Processor to Protected Mode — This function allows the user to switch the processor into protected mode. When the function call is completed, control will be transferred to an address specified by the user.

System and CPU Interrupts

In order to use this function, the user routine must create a descriptor table prior to calling the function. Each descriptor in the table must contain the limit, the base address and the access rights byte. The values in the descriptor table are used to initialize the IDTR the GDTR and the stack selector segment. The ES:SI register pair forms a pointer to the beginning of this descriptor table. Figure 7-2 illustrates the format of the descriptor table and Table 7-7 describes the descriptor table used by this function.

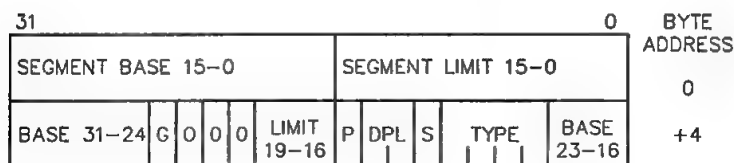


Figure 7-2. Protected Mode Descriptor Table

Table 7-7. Protected Mode Table Description

DESCRIPTOR	OFFSET	INITIALIZED BY	INITIALIZED VALUE
1 Dummy	00	User	0
2 GDT data	08	User	Data segment address
3 Interrupt	10	User	Interrupt table address
4 User data	18	User	Data segment address
5 User extra	20	User	Extra segment address
6 User stack	28	User	Stack segment address
7 Return code	30	User	User code segment
8 Code	38	Function	Function code segment

Two values are required on entry into this function, besides the ES:SI pointer value. The BL register must contain the offset value to the interrupt level 1 table. This table contains the first eight hardware interrupts. Register BH, on the other hand, contains the offset to the interrupt level 2 table. This table contains the second eight hardware interrupts.

System and CPU Interrupts

If the function executes correctly, the AH register will be set to zero. As a result of the processor mode change, all register values will change and the AX and BP register values will be destroyed. Also, because of the change in mode, BIOS calls will no longer be available to the user. This means that the user program must handle all I/O routines and instructions. The interrupt vector locations must be moved to avoid reserved memory areas of the 80386. This usually means reinitializing the interrupt hardware to the correct vectors. Also, the exception interrupt table, and the associated handler, must be initialized by the user. Finally, the interrupt descriptor table must not overlap the real mode BIOS interrupt descriptor table. This could cause unpredictable results.

Listing 7-1 provides an example of a segment of code that activates a function 89 call.

Listing 7-1. Function 89 Code Sample

```

*
*
*
SWITCH: MOV AX,GDT SEGMENT
        MOV ES,AX
        MOV SI,GDT OFFSET
        MOV BH,ILEVEL 1 OFFSET
        MOV BL,ILEVEL 2 OFFSET
        MOV AH,89H
        INT 15H
*
*
*

```

Real mode
code

Protected
mode code

Function 90H: Device Busy Loop Entry Point — This function informs the user when the ROM is about to enter a busy loop. The AL register will contain a code that identifies which system device is causing the busy condition. Table 7-8 lists the different device codes.

System and CPU Interrupts

Table 7-8. Device Type Codes

CODE	DEVICE
00H	Disk
01H	Diskette
02H	Keyboard
80H	Network
FDH	Drive motor start
FEH	Printer

Function 91H: Set Interrupt Complete Flag — This function advises the user when the ROM has completed some operation or function. The AL register will contain the appropriate code from those listed in Table 7-8.

Functions 80–91 support operations that are available on systems based on either the 80286 or 80386 processor. The extended functions, listed in Table 7-9, are only available on 80386-based computer systems.

Table 7-9. INT 15 Extended Functions

FUNCTION	DESCRIPTION
E2H	Ram protect enable/disable
E3H	Speed control
E4H	Current speed setting
E5H	Enable/disable cache memory
E6H	Read cache state

Function E2H: Ram Protect Enable/Disable — This function is used to enable and disable protection for the Monitor ROM work area. The Monitor ROM maintains a 1K memory block for various pointers and variables used during normal operations. If this area is accidentally written too by the system, unpredictable results or even a system halt could result.

System and CPU Interrupts

When entering this function, the AL register contains information on the level of protection. If the AL register is zero, the ROM will increment the protection level. If the value in AL is 10, the ROM will decrement the protection level.

Protection for this area of RAM is implemented in a stack area, similar to a last-in, first-out (LIFO) stack. The protection level is incremented or decremented depending on the value in the AL register. A byte in memory is used by the ROM to maintain this count. Whenever the value changes from 1 to 0, the RAM will be protected. If the byte changes from 0 to 1, protection is disabled. The protection state of the RAM will only be affected by transitions between the values of 0 and 1. The stack will support a maximum of 255 levels. If this value is exceeded, the byte will reset to 0 (wrap around), causing unpredictable results.

Function E3H: Set Processor Speed — This function sets the operating speed of the system processor. The AL register contains the required setting information according to the following list:

- AL = 0: Set processor to run fast at all times
- AL = -1: Set processor to run slow at all times
- AL = 1: Normally fast, but run slow for floppy disk accesses

The preferred operational mode for the 80386-based computer is fast at all times. It is possible that some applications software may have problems at the normal operating speed of this computer. In those cases you may try setting the processor to run fast except when attempting a floppy disk access. If there is a known problem with a particular package, you may have to set the processor for slow operation.

Function E4H: Read Processor Speed — This function returns the current operating speed of the system processor. The AL register contains the returned value. This value will correspond exactly with the speed codes used by function E3H.

System and CPU Interrupts

Function E5H: Enable/Disable Cache Memory — If the optional cache memory is installed in the computer, this function permits the user to selectively turn the cache on or off. The value contained in the AL register determines the on or off state of the cache. If the AL register is set to 0, the cache will be turned off. If the AL register contains a 1, the cache is turned on.

Function E6H: Read Cache State — This function returns the operational status of the cache memory, if installed, in the AL register. If bit 0 of the AL register is set, the cache is turned on. If the bit 1 of the AL register is set, a cache memory card is installed in the system. If bit 3 of the AL register is set, the cache memory failed the self-test.

NOTE: When using the extended INT 15 functions, it is a good idea to set the carry flag bit before making the function call. If you attempt a call on a computer that does not support it, the carry bit will remain set. Making an extended call on a computer that supports the function, such as the 80386-based systems, will result in a cleared carry bit.

Set/Read the Time of Day (INT 1AH)

INT 1AH is the set/read the time of day interrupt. It allows the programmer to set and read the internal clock's time and date, and to set the alarm on or off. Table 7-10 lists the different functions available under INT 1AH.

Table 7-10. INT 1AH Functions

FUNCTION	DESCRIPTION
00H	Read time of day
01H	Set time of day
02H	Read real time clock
03H	Set real time clock
04H	Read RTC date
05H	Set RTC date
06H	Set RTC alarm
07H	Disable RTC alarm

System and CPU Interrupts

Function 00H: Read Time of Day — This function allows the programmer to access the current time of day from the time and date structure. If the AL register contains a 0, then the timer count has not passed midnight since the last system reset. Otherwise, the value in the AL register will indicate the number of times the timer has passed midnight since the last system reset. The CX register contains the most-significant word in this count and the DX register contains the least-significant word.

Function 01H: Set Time of Day — This function complements function 00H. It allows the programmer to set the timer time of day. The CX and DX registers contain the same values as in function 00H.

Function 02H Read Real Time Clock — This function allows the programmer to read directly from the timer. All returned values are in BCD format. Three registers are used to return the current time:

CH:Hours
CL:Minutes
DH:Seconds

If the real time clock is not working correctly, the CF bit will be set.

Function 03H: Set Real Time Clock — This function complements function 02H, allowing the programmer to directly set the time in the real time clock. Registers CH, CL, and DH contain the same information, in the same format, as in function 02H. Register DL contains daylight saving time information. A value of 1 means that the DST option is in use; 0 means that it is not.

Function 04H: Read Real Time Clock Date — This function allows the programmer to obtain the date directly from the real time clock. All returned values are in BCD format. This function uses four registers:

CH:Century
CL:Year
DH:Month
DL:Day

If the real time clock is not operating properly, the CY bit will be set.

System and CPU Interrupts

Function 05H: Set Real Time Clock Date — This function complements the operation of function 04H. Using this function the programmer can directly set the date in the real time clock. The same registers are used here as in function 04H with the same values and formats.

Function 06H: Set Real Time Clock Alarm — This function allows the programmer to set the alarm feature of the real time clock. All values returned by this function are in BCD format. Three registers are used by this function:

CH:Hours
CL:Minutes
DH:Seconds

A value in the range of C0H – FFH in any of the registers indicates a "don't care" condition for that time value. This allows the programmer the option of setting a periodic alarm condition. Whenever the alarm conditions are satisfied, the system generates interrupt 4AH.

If the real time clock is not operating properly, the CY bit will be set.

Function 07H: Disable Real Time Clock Alarm — This function, when called, will turn off the alarm feature of the real time clock. Register AH establishes the type of operation (read or write). The CX and DX registers store the value passed to or from the 32-bit counter. Register AL serves as a flag to report that the value has rolled over into a new day since the last time the counter was read.

Most PC-compatible computers store the value that represents the time of day in a 32-bit time-of-day counter. This counter receives updates 18.2159 times a second. Therefore, 1 a.m. would be represented by a count of 65,577 and 12 noon would be 786,926. Register CX passes the high part of the count and contains the value 1 for 1 a.m. and 12 (0CH) for 12 noon. Register DX passes the low part of the count and contains 41 (29H) for 1 a.m. and 494 (1EEH) for 12 noon. The contents of register AL depends upon whether the count has rolled over into a new day since the last counter read.

System and CPU Interrupts

In this computer, a real-time clock IC keeps track of the date and time. This IC remains active even when you turn off the computer. The real-time clock uses the 32-bit time-of-day word for compatibility. When you boot the computer with MS-DOS (version 3.2 or higher), the operating system automatically reads the time of day from the clock IC. This value is then stored in the 32-bit time-of-day word. You can also use a utility program to accomplish this. The result is that software that depends on the 32-bit word will still work in this computer.

Tick Timer (INT 1CH)

INT 1CH is the tick timer interrupt. It allows you to produce CPU-independent timing loops in your programs. At powerup, this interrupt points to an IRET instruction.

The speed of the CPU clock frequency in PC-compatible computers is inconsistent from one machine to another. Furthermore, some models offer switchable-speed CPU clock frequencies. This interrupt provides a convenient means of consistently controlling timing loops regardless of the CPU clock frequency.

Each time INT 08H (the timer interrupt) executes, the tick timer interrupt is called (18.2159 times a second). Therefore, if you use INT 1CH in a program, you must save the registers at the beginning of the routine and then restore them before returning to the main program.

Alarm Interrupt (INT 4AH)

The user alarm interrupt is used whenever a real time clock alarm is active. This interrupt is generated by the computer in response to the alarm conditions specified for the real time clock. Refer to function 06H under "Set/Read the Time of Day (INT 1AH)."

If an application program uses the real time clock alarm feature, it must provide a handler for this interrupt. If no handler is provided, it will cause a system crash.

Real-Time Clock Alarm Interrupt (70H)

The INT 70H interrupt is initiated by a hardware interrupt request (IRQ8) or when an alarm from the real-time clock takes place. The alarm is set and handled through INT 15H (functions 83H and 86H). The hardware interrupt takes place approximately 1,024 times a second. Since this interrupt is essentially controlled by the hardware, it should not be called by a user program.

Programming Sound

This computer does not contain a dedicated sound chip. However, the computer can generate tones and play them through the internal speaker.

The 8254 timer is directly controllable by the programmer. The output from channel 2 of the timer is gated directly to the system speaker through a NAND gate. This gate is controlled from port B within the system. Port B is addressed within the system at hardware port address 61H. Bits 0 and 1 are responsible for turning the speaker output on and off. If these two bits contain a value of 01, the speaker will be turned off. If they contain a value of 10, the speaker will be turned on.

Programming sound, particularly music, is a specialized application and is beyond the scope of this manual. If you want to produce sounds or music, use one of the special application programs designed for that purpose or GW-BASIC, which contains statements to perform this type of task.

Chapter 8

Keyboard

This chapter describes the interrupts used for keyboard communications and lists the codes generated by the keyboard.

Programming Keyboard Interrupts

The interrupts used exclusively by the keyboard are similar to those used in previous designs. However, the 101-key keyboard supports some additional function calls that user programs can exploit. Programs designed to run on PC or PC-AT computers will function as normal. Table 8-1 indicates the keyboard interrupts for this computer.

Table 8-1. Keyboard Interrupts

INTERRUPT	FUNCTION
09H	Key pressed
16H	Keyboard input/output
1BH	Keyboard break

Key Pressed (INT 09H)

INT 09H is the key pressed interrupt which is executed each time you press or release a key. The interrupt routine reads the key from the keyboard register and encodes it, or notes the action if it is a shift or control key. The keyboard buffer stores valid key codes for access from applications programs or system software.

You should not change the action of this interrupt routine since it directly affects the action of the keyboard.

Keyboard

Keyboard Input/Output (INT 16H)

INT 16H is the keyboard input/output interrupt. This interrupt performs a keyboard operation specified by one of the function calls described in this section. The function code is placed in the AH register and then the interrupt is executed. The function codes implement:

- Key code retrieval (receive a character from the keyboard)
- Check keyboard buffer (to see if any codes are in it)
- Report keyboard status (Shift and Ctrl keys)
- Set repetition rate (for applicable keys)
- Place character in keyboard buffer (as if it were entered from the keyboard).

NOTE: The 101-key keyboard contains 17 new keys that duplicate the functions of 17 other standard keys. These new keys generate unique, extended codes. Function codes 00H-02H map these codes to the standard key codes to allow compatibility with programs written for the standard key codes. In other words, the new keys remain transparent to existing programs.

Table 8-2 is a summarized list of the INT 16H functions. A detailed discussion of each function follows the table.

Table 8-2. INT 16H Functions

FUNCTION	DESCRIPTION
00H	Get character code
01H	Check keyboard buffer
02H	Get keyboard status
03H	Set key repetition rate
05H	Place character in buffer
10H	Get extended character code
11H	Check extended keyboard buffer
12H	Get extended keyboard status

Function Code 00H: Get Character Code — This function code causes the interrupt to read a character from the keyboard buffer. When you press a key, the corresponding character code is placed in the keyboard buffer. This function will retrieve that code (ASCII codes 00H–7FH or non-ASCII codes 80H–FFH) and place it in the AL register. The code is then removed from the keyboard buffer. The AH register will contain the scan code (01H–84H) for the pressed key. If the keyboard buffer is empty, the routine waits until the next key press, generating another character code. Note that the SHIFT and CTRL keys do not generate codes, but affect the codes generated by other keys.

Function Code 01H: Check Keyboard Buffer — This function code causes the interrupt to check the status of the keyboard buffer. The zero flag (ZF) will be set if the buffer is empty. If the buffer contains key codes awaiting processing, the zero flag will be cleared (not set). The code (ASCII codes 00H–7FH or non-ASCII codes 80H–FFH) assigned to the first character in the buffer is placed in the AL register. The scan code (01H–84H) for that same key will appear in the AH register.

NOTE: This operation does not remove any codes from the keyboard buffer. Therefore, if you execute function code 01H followed by function code 00H, the same key codes will be returned. Only function code 00H removes the key codes from the buffer.

Function Code 02H: Get Keyboard Status — Two bytes of memory, 0040:0017 and 0040:0018, hold the keyboard status. This interrupt function code reports the status of the keyboard stored at location 0040:0017. Table 8-3 describes the value of this byte as stored in the AL register. If the report states that a key is pressed, that key was being held down at the time this function was executed.

Keyboard

Table 8-3. Keyboard Status Report

BIT	DESCRIPTION
0	If set (1), the right SHIFT key is pressed.
1	If set (1), the left SHIFT key is pressed.
2	If set (1), the CTRL key is pressed.
3	If set (1), the ALT key is pressed.
4	If set (1), the SCROLL LOCK mode is active.
5	If set (1), the NUM LOCK mode is active.
6	If set (1), the CAPS LOCK mode is active.
7	If set (1), the INSERT mode is active.

The keyboard status stored at location 0040:0018 can be read directly and interpreted by your own routines. Table 8-4 describes the possible values of this byte.

Table 8-4. Keyboard Status (0040:0018)

BIT	DESCRIPTION
0	Not used.
1	Not used.
2	Not used.
3	If set (1), the CTRL-NUM LOCK (pause) mode is active.
4	If set (1), the SCROLL LOCK key is pressed.
5	If set (1), the NUM LOCK key is pressed.
6	If set (1), the CAPS LOCK key is pressed.
7	If set (1), the INSERT key is pressed.

Function Code 03H: Set Key Repetition Rate — This function call will be executed when its code (03H) is placed in register AH and interrupt 16H is executed. The keyboard will respond to this interrupt and wait for a rate/delay determinant byte to be sent from the system. The rate is determined by bits 0-4 and the delay is determined by bits 5 and 6. Bit 7 is always 0. Table 8-5 indicates the repetition rates.

Table 8-5. Repetition Rates

BIT 4	BIT 3	BIT 2	BIT 1	BIT 0	REPETITION RATE
0	0	0	0	0	30.0
0	0	0	0	1	26.7
0	0	0	1	0	24.0
0	0	0	1	1	21.8
0	0	1	0	0	20.0
0	0	1	0	1	18.5
0	0	1	1	0	17.1
0	0	1	1	1	16.0
0	1	0	0	0	15.0
0	1	0	0	1	13.3
0	1	0	1	0	12.0
0	1	0	1	1	10.9
0	1	1	0	0	10.0
0	1	1	0	1	9.2
0	1	1	1	0	8.0
0	1	1	1	1	8.0
1	0	0	0	0	7.5
1	0	0	0	1	6.7
1	0	0	1	0	6.0
1	0	0	1	1	5.5
1	0	1	0	0	5.0
1	0	1	0	1	4.6
1	0	1	1	0	4.3
1	0	1	1	1	4.0
1	1	0	0	0	3.7
1	1	0	0	1	3.3
1	1	0	1	0	3.0
1	1	0	1	1	2.7
1	1	1	0	0	2.5
1	1	1	0	1	2.3
1	1	1	1	0	2.1
1	1	1	1	1	2.0

NOTE: The key repetition rate is calibrated in characters per second.

Keyboard

Function Code 05H: Place Character in Buffer — This function call is executed by placing a value of 05H in register AH and then executing interrupt 16H. This call's purpose is to allow placement of a character and associated scan code in the keyboard buffer as if it were entered from the keyboard. The character code is determined in register CL and the scan code is determined in register CH. When the INT 16H instruction occurs, these values will be placed in the keyboard buffer.

NOTE: The 101-key keyboard contains 17 new keys that duplicate the functions of 17 other standard keys. The new keys generate unique, extended codes. Function codes 10H-12H can exploit these new key codes in programs written specifically for them.

Function Code 10H: Get Extended Character Code — This function code performs the same operation as function code 00H. However, there are 17 keys that have unique codes in the 101-key keyboard. Many of these keys duplicate the function of other keys (for example, there are two sets of cursor control keys but each set generates unique code). This function code causes the interrupt to read the new key's code from the keyboard buffer.

Function Code 11H: Check Extended Keyboard Buffer — This function code places the value of any of the 17 new keys into register AL. The code (ASCII codes 00H-7FH or non-ASCII codes 80H-FFH) assigned to the first character in the keyboard buffer is placed in the AL register. The scan code (01H-84H) for that same key will appear in the AH register.

Function Code 12H: Get Extended Keyboard Status — This function code reports the status of the keyboard the same way that function code 02H does. However, function code 12H places a 2-byte status report in registers AL and AH, respectively. Table 8-6 describes the values of the byte in register AL and Table 8-7 describes the value of the byte in register AH.

Table 8-6. Keyboard Status Report (Register AL)

BIT	DESCRIPTION
0	If set (1), the right SHIFT key is pressed.
1	If set (1), the left SHIFT key is pressed.
2	If set (1), the CTRL key is active.
3	If set (1), the ALT key is active.
4	If set (1), the SCROLL LOCK mode is active.
5	If set (1), the NUM LOCK mode is active.
6	If set (1), the CAPS LOCK mode is active.
7	If set (1), the INSERT mode is active.

Table 8-7. Keyboard Status Report (Register AH)

BIT	DESCRIPTION
0	If set (1), the left CTRL key is pressed.
1	If set (1), the left ALT key is pressed.
2	If set (1), the right CTRL key is pressed.
3	If set (1), the right ALT key is pressed.
4	If set (1), the SCROLL LOCK key is pressed.
5	If set (1), the NUM LOCK key is pressed.
6	If set (1), the CAPS LOCK key is pressed.
7	System flag.

Keyboard Break (INT 1BH)

The INT 1BH instruction executes when the key pressed interrupt (09H) detects that the CTRL and BREAK keys (CTRL-BREAK) are pressed at the same time (you must press and hold the CTRL key before pressing the BREAK key).

The interrupt for this routine must exit with an IRET instruction to properly terminate the 09H interrupt. When you turn the computer on, this routine provides only the IRET instruction. That way, if a key press occurs during the self-tests, nothing will happen (unless the Esc key is pressed, which will terminate the self-tests and drive program execution to the Monitor).

Keyboard

If you allow this interrupt to retain control, you may have to accommodate one or more of the following conditions. For more information, refer to the discussion on the 8259 interrupt controller in Chapter 14.

- The break may occur during interrupt processing. You must then send one or more end-of-interrupt commands to the interrupt controller.
- If an operation was in process when you caused the interrupt to take place, all input/output devices must be reset.
- Programs that use this interrupt must not chain to the previous owner of this interrupt and you must restore the interrupt when you finish processing.

Remember, when servicing an INT 1BH instruction, your routine must perform an IRET to make sure that the interrupt restores properly. Do not link your service routine to the next one.

GW-BASIC is an example of software that uses this routine. It uses this interrupt to halt execution of a BASIC program whenever you press the CTRL-BREAK key sequence.

Keyboard Codes

The 101-key keyboard supports three different keycode sets. Set 2 is the default set. Sets 1 and 3 are optional and can be selected by sending F0H to the keyboard. The keyboard will clear its buffer and expect the system to send a byte of data representing the desired set. Byte value 01H represents code set 1 and byte value 03H represents code set 3.

In this computer, the keycodes are translated into codes used in keycode set 1 by the system control processor. The system control processor does not currently translate keycode set 3. If keycode set 3 is used, a program designed to handle this translation may have to be written. The make and break key codes for the three different code sets are provided in Tables 8-8 through 8-12.

Keyboard

Table 8-8. Alphabetic Keycode Sets

KEY	SCAN SET 1		SCAN SET 2		SCAN SET 3	
	MAKE	BREAK	MAKE	BREAK	MAKE	BREAK
A	1E	9E	1C	F0 1C	1C	F0 1C
B	30	B0	32	F0 32	32	F0 32
C	2E	AE	21	F0 21	21	F0 21
D	20	A0	23	F0 23	23	F0 23
E	12	92	24	F0 24	24	F0 24
F	21	A1	2B	F0 2B	2B	F0 2B
G	22	A2	34	F0 34	34	F0 34
H	23	A3	33	F0 33	33	F0 33
I	17	97	43	F0 43	43	F0 43
J	24	A4	3B	F0 3B	3B	F0 3B
K	25	A5	42	F0 42	42	F0 42
L	26	A6	4B	F0 4B	4B	F0 4B
M	32	B2	3A	F0 3A	3A	F0 3A
N	31	B1	31	F0 31	31	F0 31
O	18	98	44	F0 44	44	F0 44
P	19	99	4D	F0 4D	4D	F0 4D
Q	10	90	15	F0 15	15	F0 15
R	13	93	2D	F0 2D	2D	F0 2D
S	1F	9F	1B	F0 1B	1B	F0 1B
T	14	94	2C	F0 2C	2C	F0 2C
U	16	96	3C	F0 3C	3C	F0 3C
V	2F	AF	2A	F0 2A	2A	F0 2A
W	11	91	1D	F0 1D	1D	F0 1D
X	2D	AD	22	F0 22	22	F0 22
Y	15	95	35	F0 35	35	F0 35
Z	2C	AC	1A	F0 1A	1A	F0 1A

NOTE: All values are expressed in hexadecimal.

Keyboard

Table 8-9. Numeric and Punctuation Key Scan Code Sets

KEY	SCAN SET 1		SCAN SET 2		SCAN SET 3	
	MAKE	BREAK	MAKE	BREAK	MAKE	BREAK
1/!	02	82	16	F0 16	16	F0 16
2/@	03	83	1E	F0 1E	1E	F0 1E
3/#	04	84	26	F0 26	26	F0 26
4/\$	05	85	25	F0 25	25	F0 25
5/%	06	86	2E	F0 2E	2E	F0 2E
6/^	07	87	36	F0 36	36	F0 36
7/&	08	88	3D	F0 3D	3D	F0 3D
8/*	09	89	3E	F0 3E	3E	F0 3E
9/(0A	8A	46	F0 46	46	F0 46
0/)	0B	8B	45	F0 45	45	F0 45
-/_	0C	8C	4E	F0 4E	4E	F0 4E
=/+	0D	8D	55	F0 55	55	F0 55
'/~	29	A9	0E	F0 0E	0E	F0 0E
[{	1A	9A	54	F0 54	54	F0 54
]}	1B	9B	5B	F0 5B	5B	F0 5B
:/	27	A7	4C	F0 4C	4C	F0 4C
'"	28	A8	52	F0 52	52	F0 52
,/<	33	B3	41	F0 41	41	F0 41
./>	34	B4	49	F0 49	49	F0 49
//?	35	B5	4A	F0 4A	4A	F0 4A
\	2B	AB	5D	F0 5D	5C	F0 5C

NOTE: All values are expressed in hexadecimal.

Table 8-10. Function and Control Key Scan Code Sets

KEY	SCAN SET 1		SCAN SET 2		SCAN SET 3	
	MAKE	BREAK	MAKE	BREAK	MAKE	BREAK
F1	3B	BB	05	F0 05	07	F0 07
F2	3C	BC	06	F0 06	0F	F0 0F
F3	3D	BD	04	F0 04	17	F0 17
F4	3E	BE	0C	F0 0C	1F	F0 1F
F5	3F	BF	03	F0 03	27	F0 27
F6	40	C0	0B	F0 0B	2F	F0 2F
F7	41	C1	83	F0 83	37	F0 37
F8	42	C2	0A	F0 0A	3F	F0 3F
F9	43	C3	01	F0 01	47	F0 47
F10	44	C4	09	F0 09	4F	F0 4F
F11	57	D7	78	F0 78	56	F0 56
F12	58	D8	07	F0 07	5E	F0 5E
BACK SPACE	0E	8E	66	F0 66	66	F0 66
TAB	0F	8F	0D	F0 0D	0D	F0 0D
CAPS LOCK	3A	BA	58	F0 58	14	F0 14
SCROLL LOCK	46	C6	7E	F0 7E	5F	F0 5F
ENTER/ RETURN	1C	9C	5A	F0 5A	5A	F0 5A
Left SHIFT	2A	AA	12	F0 12	12	F0 12
Right SHIFT	36	B6	59	F0 59	59	F0 59
Space bar	39	B9	29	F0 29	29	F0 29
ESC	01	81	76	F0 76	08	F0 08
Left CTRL	1D	9D	14	F0 14	11	F0 11

Keyboard

Table 8-10 (continued). Function and Control Key Scan Code Sets

KEY	SCAN SET 1		SCAN SET 2		SCAN SET 3	
	MAKE	BREAK	MAKE	BREAK	MAKE	BREAK
Left ALT	38	B8	11	F0 11	19	F0 19
Right CTRL	E0 1D	E0 9D	E0 14	E0 F0 1458		F0 58
Right ALT	E0 38	E0 B8	E0 11	E0 F0 1139		F0 39

NOTE: All values are expressed in hexadecimal.

Table 8-11. Factored Keypad Scan Code Sets

KEY	SHIFT ACTIVE		NUM LOCK ACTIVE	
	MAKE / BREAK	MAKE / BREAK	MAKE / BREAK	MAKE / BREAK
Scan Code Set 1				
INSERT	E0H 52H/ E0H D2H	E0H AAH E0H 52H/ E0H D2H E0H 2AH	E0H 2AH E0H 52H/ E0H D2H E0H AAH	
DELETE	E0H 53H/ E0H D3H	E0H AAH E0H 53H/ E0H D3H E0H 2AH	E0H 2AH E0H 53H/ E0H D3H E0H AAH	
←	E0H 4BH/ E0H CBH	E0H AAH E0H 4BH/ E0H CBH E0H 2AH	E0H 2AH E0H 4BH/ E0H CBH E0H AAH	
HOME	E0H 47H/ E0H C7H	E0H AAH E0H 47H/ E0H C7H E0H 2AH	E0H 2AH E0H 47H/ E0H C7H E0H AAH	
END	E0H 4FH/ E0H CFH	E0H AAH E0H 4FH/ E0H CFH E0H 2AH	E0H 2AH E0H 4FH/ E0H CFH E0H AAH	
Up arrow	E0H 48H/ E0H C8H	E0H AAH E0H 48H/ E0H C8H E0H 2AH	E0H 2AH E0H 48H/ E0H C8H E0H AAH	

Table 8-11 (continued). Factored Keypad Scan Code Sets

KEY	SHIFT ACTIVE		NUM LOCK ACTIVE	
	MAKE / BREAK	MAKE / BREAK	MAKE / BREAK	
Down arrow	E0H 50H/ E0H D0H	E0H AAH E0H 50H/ E0H D0H E0H 2AH	E0H 2AH E0H 50H/ E0H D0H E0H AAH	
PAGE UP	E0H 49H/ E0H C9H	E0H AAH E0H 49H/ E0H C9H E0H 2AH	E0H 2AH E0H 49H/ E0H C9H E0H AAH	
PAGE DN	E0H 51H/ E0H D1H	E0H AAH E0H 51H/ E0H D1H E0H 2AH	E0H 2AH E0H 51H/ E0H D1H E0H AAH	
→	E0H 4DH/ E0H CDH	E0H AAH E0H 4DH/ E0H CDH E0H 2AH	E0H 2AH E0H 4DH/ E0H CDH E0H AAH	
PRINT SCREEN	E0 2A E0 37/ E0 B7 E0 AA	E0 37/ E0 B7 (see note 2)		
PAUSE/ BREAK	(see note 3)			

Scan Code Set 2

INSERT	E0 70/ E0 F0 70	E0 F0 12 E0 70/ E0 F0 70 E0 12	E0 12 E0 72/ E0 F0 70 E0 F0 12
DELETE	E0 71/ E0 F0 71	E0 F0 12 E0 71/ E0 F0 71 E0 12	E0 12 E0 71/ E0 F0 71 E0 F0 12
←	E0 6B/ E0 F0 6B	E0 F0 12 E0 6B/ E0 F0 6B E0 12	E0 12 E0 6B/ E0 F0 6B E0 F0 12
HOME	E0 6C/ E0 F0 6C	E0 F0 12 E0 6C/ E0 F0 6C E0 12	E0 12 E0 6C/ E0 F0 6C E0 F0 12
END	E0 69/ E0 F0 69	E0 F0 12 E0 69/ E0 F0 69 E0 12	E0 12 E0 69/ E0 F0 69 E0 F0 12
83	E0 75/ E0 F0 75	E0 F0 12 E0 75/ E0 F0 75 E0 12	E0 12 E0 75/ E0 F0 75 E0 F0 12

Keyboard

Table 8-11 (continued). Factored Keypad Scan Code Sets

KEY	SHIFT ACTIVE		NUM LOCK ACTIVE	
	MAKE / BREAK	MAKE / BREAK	MAKE / BREAK	
84	E0 72/ E0 F0 72	E0 F0 12 E0 72/ E0 F0 72 E0 12	E0 12 E0 72/ E0 F0 72 E0 F0 12	
PAGE UP	E0 7D/ E0 F0 7D	E0 F0 12 E0 7D/ E0 F0 7D E0 12	E0 12 E0 7D/ E0 F0 7D E0 F0 12	
PAGE DOWN	E0 7A/ E0 F0 7A	E0 F0 12 E0 7A/ E0 F0 7A E0 12	E0 12 E0 7A/ E0 F0 7A E0 F0 12	
→	E0 74/ E0 F0 74	E0 F0 12 E0 74/ E0 F0 74 E0 12	E0 12 E0 74/ E0 F0 74 E0 F0 12	
PRINT SCREEN	(see note 4)			
PAUSE/ BREAK	(see note 5)			

Scan Code Set 3

INSERT	67 F0 67	(see note 6)	(see note 6)
DELETE	64 F0 64		
←	61 F0 61		
HOME	6E F0 6E		
END	65 F0 65		
83	63 F0 63		
84	60 F0 60		
PAGE UP	6F F0 6F		
PAGE DOWN	6D F0 6D		
→	6A F0 6A		

Table 8-11 (continued). Factored Keypad Scan Code Sets

KEY	SHIFT ACTIVE		NUM LOCK ACTIVE
	MAKE / BREAK	MAKE / BREAK	MAKE / BREAK
PRINT SCREEN	57 F0 57		
PAUSE/ BREAK	62 F0 62		

NOTES:

1. All values are expressed in hexadecimal.
2. In scan code set 1 the Print Screen key generates E0 37/E0 B7 when the Shift key or the CTRL key is active. It also generates make/break code 54/D4 when the ALT key is active.
3. This key generates E1 1D 45 E1 9D C5 when pressed. It does not generate a code when released. When the CTRL key is active E0 46 E0 C6 is generated.
4. In scan code set 2 the PRINT SCREEN key generates E0 12 E0 7C/E0 F0 7C E0 F0 12. When the SHIFT key or the CTRL key is active it generates make/break code E0 7C/E0 F0 7C. When the ALT key is active it generates 84/F0 84.
5. This key generates E1 14 77 E1 F0 14 F0 77 when pressed. It does not generate a code when released. When the CTRL key is active E0 7E E0 F0 7E is generated.
6. The same codes are generated when the SHIFT or NUM LOCK keys are active.

Table 8-12. Numeric Keycode Sets

KEY	SCAN SET 1		SCAN SET 2		SCAN SET 3	
	MAKE	BREAK	MAKE	BREAK	MAKE	BREAK
NUM LOCK	45	C5	77	F0 77	76	F0 76
1/END	4F	CF	69	F0 69	69	F0 69
2/	50	D0	72	F0 72	72	F0 72
3/PGDN	51	D1	7A	F0 7A	7A	F0 7A
4/←	4B	CB	6B	F0 6B	6B	F0 6B
5/	4C	CC	73	F0 73	73	F0 73

Keyboard

Table 8-12 (continued). Numeric Keycode Sets

KEY	SCAN SET 1		SCAN SET 2		SCAN SET 3	
	MAKE	BREAK	MAKE	BREAK	MAKE	BREAK
6/	4D	CD	74	F0 74	74	F0 74
7/HOME	47	C7	6C	F0 6C	6C	F0 6C
8/	48	C8	75	F0 75	75	F0 75
9/PGUP	49	C9	7D	F0 7D	7D	F0 7D
0/INSERT	52	D2	70	F0 70	70	F0 70
./DEL	53	D3	71	F0 71	71	F0 71
+	4E	CE	79	F0 79	7C	F0 7C
-	4A	CA	7B	F0 7B	84	F0 84
*	37	B7	7C	F0 7C	7E	F0 7E
ENTER	E0 1C	E0 9C	E0 5A	E0 F0 5A	79 F0 79	

NOTES:

In code set 1, the / key generates make/break code E0 35/E0 B5 and also generates E0 AA E0 35/E0 B5 E0 2A when the SHIFT key is active. In scan code set 2, the key generates E0 4A/E0 F0 4A and, when the SHIFT key is active, it generates E0 F0 12 4A/ E0 12 F0 4A. In scan code set 3, the key generates 77/F0 77.

All values are expressed in hexadecimal.

The following sections describe the system scan codes. A scan code refers to the hexadecimal value that is generated when a key is pressed. Scan codes consist of the key's make/break code in conjunction with a standard ASCII code. The scan codes that are generated when the SHIFT, CTRL, and ALT keys are active are also provided. The make/break codes for each code set (1, 2, and 3) are provided in Tables 8-7 through 8-11.

System Scan Codes

The 101-key keyboard keys are configured in a matrix that consists of scan lines and sense lines. Each intersection point of the scan and sense lines represents a key location. The following sections are grouped into logical divisions according to key groups on the keyboard. Figure 8-1 illustrates the 101-key keyboard layout.

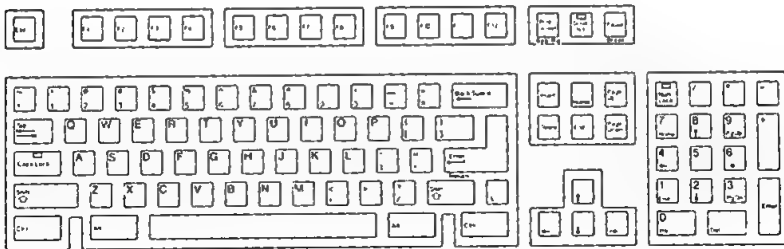


Figure 8-1. 101-Key Keyboard

Alphabetic Keys

The 26 alphabetic keys have the standard QWERTY typewriter arrangement and are identified in Figure 8-2. The SHIFT keys and the CAPS LOCK key affect the function of the alphabetic keys as on a typewriter. The exception is that the CAPS LOCK key affects only the alphabetic keys. All other keys are unaffected by its action.

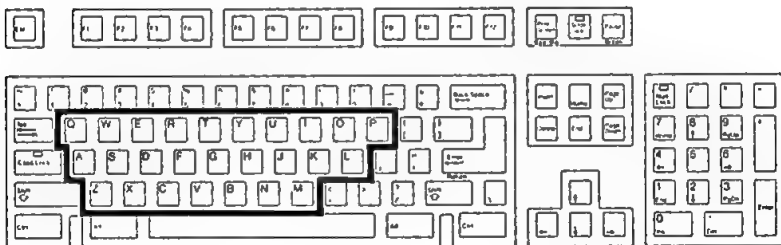


Figure 8-2. Alphabetic Keys

Keyboard

When a key is pressed, a key code representing that particular key is generated by the keyboard and sent to the computer. The operating system (and Monitor program) use the code in conjunction with standard ASCII values to generate scan codes. The scan codes can be affected by the active states of the SHIFT, CTRL, and ALT keys. Application programs are written to interpret the scan codes to execute some task.

When a key is released, a key "break" code is generated. Application programs can also be written to interpret these codes to execute some task.

Table 8-13 details the key closure (make) codes, ASCII codes, and scan codes generated for the alphabetic keys. The scan codes produced by the active SHIFT, CTRL, and ALT keys are also provided.

Table 8-13. Alphabetic Key Codes (AT Mode)

KEY	MAKE CODE	ASCII CODE	SCAN CODE	KEY PRESSED		
				SHIFT	CTRL	ALT
A	1EH	61H	1E61H	1E41H	1E01H	1E00H
B	30H	62H	3062H	3042H	3002H	3000H
C	2EH	63H	2E63H	3E43H	2E03H	2E00H
D	20H	64H	2064H	2044H	2004H	2000H
E	12H	65H	1265H	1245H	1205H	1200H
F	21H	66H	2166H	2146H	2106H	2100H
G	22H	67H	2267H	2247H	2207H	2200H
H	23H	68H	2368H	2348H	2308H	2300H
I	17H	69H	1769H	1749H	1709H	1700H
J	24H	6AH	246AH	244AH	240AH	2400H
K	25H	6BH	256BH	254BH	250BH	2500H
L	26H	6CH	266CH	264CH	260CH	2600H
M	32H	6DH	326DH	324DH	320DH	3200H
N	31H	6EH	316EH	314EH	310EH	3100H
O	18H	6FH	186FH	184FH	180FH	1800H
P	19H	70H	1970H	1950H	1910H	1900H
Q	10H	71H	1071H	1051H	1011H	1000H
R	13H	72H	1372H	1352H	1312H	1300H
S	1FH	73H	1F73H	1F53H	1F13H	1F00H
T	14H	74H	1474H	1454H	1414H	1400H
U	16H	75H	1675H	1655H	1615H	1600H

Table 8-13 (continued). Alphabetic Key Codes (AT Mode)

KEY	MAKE CODE	ASCII CODE	SCAN CODE	KEY PRESSED		
				SHIFT	CTRL	ALT
V	2FH	76H	2F76H	2F56H	2F16H	2F00H
W	11H	77H	1177H	1157H	1117H	1100H
X	2DH	78H	2D78H	2D58H	2D18H	2D00H
Y	15H	79H	1579H	1559H	1519H	1500H
Z	2CH	7AH	2C7AH	2C5AH	2C1AH	2C00H

Numeric and Punctuation Keys

The numeric and punctuation keys, shown in Figure 8-3, include the numbers 0 through 9, the common punctuation marks, and the special programming characters that make up the remainder of the printable ASCII character set. Each of these keys can generate two characters. The uppercase character is generated when you press either SHIFT key in combination with another key. Table 8-14 details the make codes, ASCII codes, and scan codes generated for the numeric and punctuation keys.

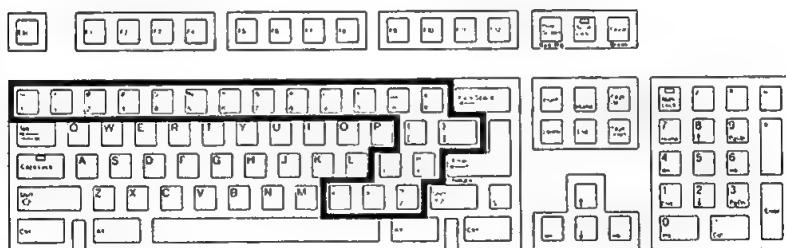


Figure 8-3. Numeric and Punctuation Keys

Keyboard

Table 8-14. Numeric and Punctuation Key Codes

KEY	MAKE CODE	ASCII CODE	SCAN CODE	KEY PRESSED		
				SHIFT	CTRL	ALT
! / 1	02H	31H	0231H	0221H		7800H
@ / 2	03H	32H	0332H	0340H	0300H	7900H
# / 3	04H	33H	0433H	0423H		7A00H
\$ / 4	05H	34H	0534H	0524H		7B00H
% / 5	06H	35H	0635H	0625H		7C00H
↑ / 6	07H	36H	0736H	075EH	071EH	7D00H
& / 7	08H	37H	0837H	0826H		7E00H
* / 8	09H	38H	0938H	092AH		7F00H
(/ 9	0AH	39H	0A39H	0A28H		8000H
) / 0	0BH	30H	0B30H	0B29H		8100H
_ / -	0CH	2DH	0C2DH	0C5FH	0C1FH	8200H
+ / =	0DH	3DH	0D3DH	0D2BH	8300H	
~ / ' `	29H	60H	2960H	297EH		
{ / [1AH	5BH	1A5BH	1A7BH	1A1BH	
} /]	1BH	5DH	1B5DH	1B7DH	1B1DH	
: / ;	27H	3BH	273BH	273AH		
" / "	28H	27H	2827H	2822H		
< / ,	33H	2CH	332CH	333CH		
> / .	34H	2EH	342EH	343EH		
? / /	35H	2FH	352FH	353FH		
/ \	2BH	5CH	2B5CH	2B7CH	2B1CH	

Function and Control Keys

The function and control keys, shown in Figure 8-4, do not generate their own codes but modify the codes produced by other keys. They are described in Table 8-15 and their scan codes are listed in Table 8-16.

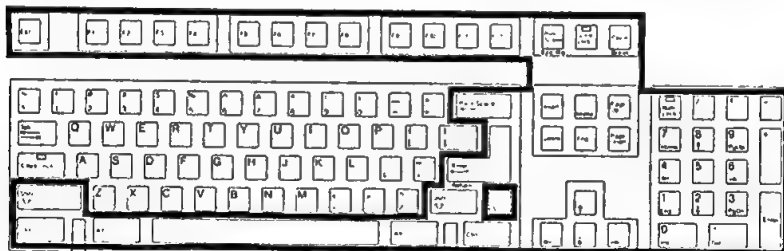


Figure 8-4. Function and Control Keys

Table 8-15. Function and Control Keys

KEY	DESCRIPTION
Left ALT	Alternate. In addition to modifying the codes produced by other keys, the ALT key can be used to generate any hexadecimal code from 0 to FFH (0 to 255 decimal). The values are displayed as an equivalent ASCII character. Lock the keypad, press and hold either ALT key, and then enter the decimal value through the keypad. When you release the ALT key, the keyboard controller generates the hexadecimal value and the system displays it on the video monitor.
Right ALT	Alternate. This key generates unique make and break codes (and therefore unique scan codes). Programs can be written to interpret the unique codes and execute a specific function. To allow compatibility with existing programs, function calls 00H-02H remove the unique code characteristics from the right ALT key. It then functions like the left ALT key.
BACK SPACE	Back space. Normally generates ASCII code 08H.
BREAK	Break. Normally used to break program execution. Press and hold the SHIFT key and then press the BREAK key.
CAPS LOCK	Caps Lock. Toggles the CAPS LOCK mode. The SHIFT keys reverse the action of the CAPS LOCK mode.
Left CTRL	Control. Modifies the codes produced by other keys by adding a special hexadecimal prefix code to the keys individual code. Applications recognize the additional code and respond as required.
Right CTRL	Control. This key generates unique make and break codes (and therefore unique scan codes). Programs can be written to interpret the unique codes and execute a specific function. To allow compatibility with existing programs, function calls 00H-02H remove the unique code characteristics from the right CTRL key. It then functions like the left CTRL key.

Keyboard

Table 8-15 (continued). Function and Control Keys

KEY	DESCRIPTION
Cursor keys	Cursor/screen control. Software uses the codes produced by these keys to move the cursor up, down, left, right, and control screen movement. The separate (gray) cursor control keys generate unique make and break codes. Software can be written to exploit these keys or a function call (00H-02H) can be used to provide the same functions as the cursor control keys on the numeric keypad.
DEL	Delete. The code produced by the DEL key is used by most text editing and word processing software to delete material. It is also used with the CTRL and ALT keys to reset the computer. The separate (gray) DELETE key generates unique make and break codes. Software can be written to exploit these codes or a function call (00H-02H) can be used to provide the same functions as the Del key on the numeric keypad.
END	End. The code produced by the END key is used by most text editing and word processing software for screen, page, or document manipulation. The separate (gray) END key generates unique make and break codes. Software can be written to exploit these codes or a function call (00H-02H) can be used to provide the same functions as the END key on the numeric keypad.
ESC	Escape. Normally generates ASCII code 1BH.
F1-F12	Function keys. The codes produced by these keys work with different software products that generate user-defined functions.
HOME	Home. The code produced by the HOME key is used by most text editing and word processing software for screen, page, or document manipulation. The separate (gray) HOME key generates unique make and break codes. Software can be written to exploit these codes or a function call (00H-02H) can be used to provide the same functions as the HOME key on the numeric keypad.

Table 8-15 (continued). Function and Control Keys

KEY	DESCRIPTION
NUM LOCK	Numbers Lock. When the computer is powered up, the keypad is active. This key toggles the keypad between numeric and cursor/screen control modes. The SHIFT keys reverse the numeric and cursor/screen control modes.
PGDN	Page Down. The code produced by the PGDN key is used by most text editing and word processing software for screen, page, or document manipulation. The separate (gray) PAGE DOWN key generates unique make and break codes. Software can be written to exploit these codes or a function call (00H-02H) can be used to provide the same functions as the PGDN key on the numeric keypad.
PGUP	Page Up. The code produced by the PGUP key is used by most text editing and word processing software for screen, page, or document manipulation. The separate (gray) PAGE UP key generates unique make and break codes. Software can be written to exploit these codes or a function call (00H-02H) can be used to provide the same functions as the PGUP key on the numeric keypad.
PRINT SCREEN	Print Screen. Sends the contents of the screen to the LPT1 device. If the operating system is configured to map the parallel port to the serial port (COM1), the screen contents is routed through COM1.
ENTER/RETURN	Enter/Return. Normally generates ASCII code 0DH. Software usually adds ASCII code 0AH.
SCROLL LOCK	Scroll Lock. The code produced by this key is used by software to control screen scrolling.
SHIFT	Shift. Modifies the codes produced by other keys. Reverses the action of CAPS LOCK and NUM LOCK.
Space bar	Space. Normally generates ASCII code 20H.
TAB	Tab. Normally generates ASCII code 09H.

Keyboard

Table 8-16. Function and Control Key Scan Codes

KEY	MAKE CODE	ASCII CODE	SCAN CODE	KEY PRESSED		
				SHIFT	CTRL	ALT
F1	3BH	00H	3B00H	5400H	5E00H	6800H
F2	3CH	00H	3C00H	5500H	5F00H	6900H
F3	3DH	00H	3D00H	5600H	6000H	6A00H
F4	3EH	00H	3E00H	5700H	6100H	6B00H
F5	3FH	00H	3F00H	5800H	6200H	6C00H
F6	40H	00H	4000H	5900H	6300H	6D00H
F7	41H	00H	4100H	5A00H	6400H	6E00H
F8	42H	00H	4200H	5B00H	6500H	6F00H
F9	43H	00H	4300H	5C00H	6600H	7000H
F10	44H	00H	4400H	5D00H	6700H	7100H
F11	57H	00H	8500H	8700H	8900H	8B00H
F12	58H	00H	8600H	8800H	8A00H	8C00H
BACK SPACE	0EH	08H	0E08H	0E08H	0E7FH	0E00H
ESC	01H	1BH	011BH	011BH	011BH	0100H
RETURN	1CH	0DH	1C0DH	1C0DH	1C0AH	1C00H
Space bar	39H	20H	3920H	3920H	3920H	3920H
7/HOME	47H	2DH	4737H	4700H	7700H	0007H
HOME	*	E0H	47E0H	47E0H	77E0H	97E0H
9/PGUP	49H	2DH	4939H	4900H	8400H	9900H
PAGE UP	*	E0H	49E0H	49E0H	84E0H	0009H
1/END	4FH	00H	4F31H	4F00H	7500H	9F00H
END	*	E0H	4FE0H	4FE0	75E0H	0001H
3/PGDN	51H	00H	5133H	5100H	7600H	
PAGE DOWN	*	E0H	51E0H	51E0H	76E0H	A1E0H
./DEL	53H	00H	532EH	5300H	9300H	
DELETE	*	E0H	53E0H	53E0H	93E0H	A300H
TAB	0FH	09H	0F09H		9400H	A500H
PRINT SCREEN					7200H	

Table 8-16 (continued). Function and Control Key Scan Codes

KEY	MAKE CODE	ASCII CODE	SCAN CODE	KEY PRESSED		
				SHIFT	CTRL	ALT

NOTES:

The scan code for the majority of keys consist of each key's make code and its standard ASCII code. Some keys generate different scan codes with respect to their make codes, however.

The codes marked with an asterisk are supplied in Table 8-15.

The following keys do not generate unique scan codes but can modify the values of other scan codes:

ALT
CAPS LOCK
CTRL
NUM LOCK
PAUSE
SHIFT
SCROLL LOCK
BREAK

Make/Break Key Codes

Each time you press a key, it produces two distinct codes: the make code when the key is pressed and the break code when you release the key. One or more bytes are produced by the keyboard and interpreted by INT 09H. These codes are distinct from the scan codes resulting from the use of the INT 16H function. To access the make/break codes, INT 09H has to be modified. Do not alter this interrupt unless you are an experienced programmer. Table 8-17 lists all the make/break codes for scan set 2.

Keyboard

Table 8-17. Make/Break Key Codes

KEY	MAKE	BREAK
A	1EH	9EH
B	30H	B0H
C	2EH	AEH
D	20H	A0H
E	12H	92H
F	21H	A1H
G	22H	A2H
H	23H	A3H
I	17H	97H
J	24H	A4H
K	25H	A5H
L	26H	A6H
M	32H	B2H
N	31H	B1H
O	18H	98H
P	19H	99H
Q	10H	90H
R	13H	93H
S	1FH	9FH
T	14H	94H
U	16H	96H
V	2FH	AFH
W	11H	91H
X	2DH	ADH
Y	15H	95H
Z	2CH	ACH
1/I	02H	82H
2/@	03H	83H
3/#	04H	84H
4/\$	05H	85H
5/%	06H	86H
6/↑	07H	87H
7/&	08H	88H
8/*	09H	89H
9/(0AH	8AH
0/)	0BH	8BH
-/_	0CH	8CH
=/+	0DH	8DH
'/>	29H	A9H
[/{	1AH	9AH
]}/	1BH	9BH

Table 8-17 (continued). Make/Break Key Codes

KEY	MAKE	BREAK
;/:	27H	A7H
'/	28H	A8H
,/<	33H	B3H
./>	34H	B4H
//?	35H	B5H
\/	2BH	ABH
F1	3BH	BBH
F2	3CH	BCH
F3	3DH	BDH
F4	3EH	BEH
F5	3FH	BFH
F6	40H	COH
F7	41H	CIH
F8	42H	C2H
F9	43H	C3H
F10	44H	C4H
F11	57H	D7H
F12	58H	D8H
BACK SPACE	0EH	8EH
TAB	0FH	8FH
CAPS LOCK	3AH	BAH
SCROLL LOCK	46H	C6H
ENTER/RETURN	1CH	9CH
Left SHIFT	2AH	AAH
Right SHIFT	36H	B6H
Space bar	39H	B9H
ESC	01H	81H
Left CTRL	1DH	9DH
Left ALT	38H	B8H
Right CTRL	E0H 1DH	E0H 9DH
Right ALT	E0H 38H	E0H B8H
PAUSE	E1H 1DH 45H	E1H 9DH C5H
PRINT SCREEN	E0H 2AH E0H 37H	E0H B7H E0H AAH

Keyboard

Table 8-17 (continued). Make/Break Key Codes

KEY	MAKE	BREAK
Factored Keypad		
INSERT	E0H 2AH E0H 52H	E0H D2H E0H AAH
HOME	E0H 2AH E0H 47H	E0H C7H E0H AAH
PAGE UP	E0H 2AH E0H 49H	E0H C9H E0H AAH
DELETE	E0H 2AH E0H 53H	E0H D3H E0H AAH
END	E0H 2AH E0H 4FH	E0H CFH E0H AAH
PAGE DOWN	E0H 2AH E0H 51H	E0H D1H E0H AAH
Up arrow	E0H 2AH E0H 48H	E0H C8H E0H AAH
Left arrow	E0H 2AH E0H 4BH	E0H CBH E0H AAH
Down arrow	E0H 2AH E0H 50H	E0H D0H E0H AAH
Right arrow	E0H 2AH E0H 4DH	E0H CDH E0H AAH
Number Keypad		
1	4FH	CFH
2	50H	DFH
3	51H	D1H
4	4BH	CBH
5	4CH	CCH
6	4DH	CDH
7	47H	C7H
8	48H	C8H
9	49H	C9H
NUM LOCK	45H	C5H
*	37H	B7H
-	4AH	CAH
+	4EH	CEH
INS	52H	D2H
DEL	53H	D3H
ENTER	E0H 1CH	E0H 9CH
/	E0H 35H	E0H B5H

Chapter 9

Input/Output Interrupts

This chapter describes the interrupts used for communications, printers, and serial and parallel communications.

Programming Input/Output Interrupts

Table 9-1 defines the input/output interrupts described in this chapter. Refer to Chapter 6 for information on how to use and program these interrupts.

Table 9-1. Input/Output Interrupts

INTERRUPT	FUNCTION
05H	Print screen
0BH	Communications (COM2)
0CH	Communications (COM1)
0DH	Alternate parallel printer (LPT2)
0FH	Parallel printer (LPT1)
14H	Serial input/output
17H	Printer input/output
18H	Parallel/serial configuration

Print Screen (INT 05H)

INT 05H sends the contents of the screen to the LPT device (usually a printer). Valid characters, or codes that match a valid character, are sent to the printer. All other shapes will be ignored by the default print screen routine. This interrupt routine performs the same function as when you press the PRINT SCREEN key.

To aid in using the print screen function, the system reserves a byte of memory at location 0050:0000 as a status byte. While the print screen routine is executing, this byte is set to 01H. This serves

Input/Output Interrupts

as a flag to prevent additional print screen requests from being processed while the current routine is executing. Upon completion of the print screen routine, the status byte changes to 00H if no errors occurred or FFH if an error did occur. The error is usually caused by a printer timeout.

Communications (INT 0BH and INT 0CH)

The INT 0BH and INT 0CH instructions provide serial communications capabilities through COM1 (INT 0CH) and COM2 (INT 0BH). Note that COM2 is not supplied in the basic system configuration. The second port can be used if it is supplied on a supplemental controller card. These interrupts are normally reserved for use with applications software. If you want to use these interrupts for your own communication activities, you must supply your own input/output routines.

NOTE: Interrupts INT 14H and INT 18H provide support for Monitor program and MS-DOS serial communications.

Parallel Printer (INT 0DH and INT 0FH)

INT 0FH (LPT1) and INT 0DH (LPT2) are the parallel communications interrupts, generally used with printers. These interrupts are normally reserved for use with applications software. If you want to use these interrupts, you must supply your own input/output routines.

NOTE: INT 17H provides complete support for parallel communications.

Serial Input/Output (INT 14H)

INT 14H allows you to perform the serial functions described in Table 9-2 and explained in the following paragraphs. You must place a specific function code in the AH register before you can execute this interrupt. Register DX must be loaded with 0 for COM1 or 1 for COM2. For more information, refer to "Parallel/Serial Configuration" in this chapter.

Input/Output Interrupts

NOTE: The addressed serial device must be connected to the system. Since the Advanced Desktop computer has only one serial port, always set the DX register to 0 for COM1.

Table 9-2. Serial Input/Output Function Codes

CODE	DESCRIPTION
00H	Initialize the serial input/output port
01H	Send character to the serial port
02H	Receive character from the serial port
03H	Read communications status

Function Code 00H: Initialize the Serial Input/Output Port — This function code initializes the parameters of the serial port according to the value in the AL register. Table 9-3 through Table 9-6 define the possible values for this code. Register DX must be loaded with 0 for the COM1 port.

Table 9-3. Mode-Select Byte Breakdown

BIT	DESCRIPTION
0-1	Sets word length (refer to Table 9-4)
2	Sets number of stop bits: 0 = 1 stop bit, 1 = 2 stop bits
3-4	Sets parity selection (refer to Table 9-5)
5-7	Sets baud rate (refer to Table 9-6).

Table 9-4. Word Length Selection

BIT 1	BIT 0	WORD LENGTH
0	0	5 bits (not supported in PC-compatible computers)
0	1	6 bits (not supported in PC-compatible computers)
1	0	7 bits
1	1	8 bits

NOTE: When you transmit or receive a word with a length of less than eight bits, the character will be converted by the asynchronous communications element. The asynchronous communications element will convert a transmitted word from a full 8-bit word to the specified length. A received word will be converted from the received length to a full 8-bit word. Extra bits will be ignored or set to 0.

Input/Output Interrupts

Table 9-5. Parity Selection

BIT 4	BIT 3	SELECTION
0	0	No parity
0	1	Odd parity
1	0	No parity
1	1	Even parity

Table 9-6. Baud Rate Selection

BIT 7	BIT 6	BIT 5	BAUD RATE
0	0	0	110
0	0	1	150
0	1	0	300
0	1	1	600
1	0	0	1200
1	0	1	2400
1	1	0	4800
1	1	1	9600

NOTE: Although the hardware is capable of handling other baud rates, these are the only rates supported by the interrupt.

Upon return from the routine, the 16-bit AX register will contain the serial port status report. The eight bits of register AX that make up register AH will contain the line control status (refer to Table 9-7). The eight bits of register AX that make up register AL will contain the modem control status (refer to Table 9-8). Function code 03H of this interrupt will return the same report.

Function Code 01H: Send Character to the Serial Port — This function code transmits the byte stored in the AL register out the specified serial port. Register DX must contain a 0 for the COM1 port.

Input/Output Interrupts

Upon return from the routine, the 16-bit AX register will contain the serial port status report. The eight bits of register AX that make up register AH will contain the line control status (refer to Table 9-7). The eight bits of register AX that make up register AL will contain the modem control status (refer to Table 9-8). Function code 03H of this interrupt will return the same report.

NOTE: In place of the timeout error indicated in Table 9-7, the most-significant bit of register AH will contain the transmit status bit. If the routine cannot transmit the character placed in register AL for some reason, the transmit status bit will be set (1).

Function Code 02H: Receive Character from the Serial Port — This function code loads the byte at the specified serial port into the AL register. Register DX must contain a 0 for the COM1 port.

Upon return from the routine, the 16-bit AX register will contain the serial port status report. The eight bits of register AX that make up register AH will contain the line control status (refer to Table 9-7). The eight bits of register AX that make up register AL will contain the modem control status (refer to Table 9-8). Function code 03H of this interrupt will return the same report.

NOTE: Bits 7, 4, 3, 2, and 1 of register AH will contain the data transfer status (refer to Table 9-7). If the contents of register AH is 0, the routine has read the byte properly into AL. If AH is not 0, some type of error occurred. In these cases, time-out errors refer to either the absence of either the data set ready (DSR) signal or the clear to send (CTS) signal.

Function Code 03H: Read Communications Status — This function code will return a report of the status of the specified communications port. Register DX must contain a 0 for the COM1 port.

The status report returns as a bit-mapped value in register AX. The eight bits of register AX that make up register AH contain the line control status (refer to Table 9-7). The eight bits of register AX that make up register AL contain the modem control status (refer to Table 9-8).

Input/Output Interrupts

Table 9-7. Line Control Status

BIT	DESCRIPTION
0	When set (1), the received data is ready.
1	When set (1), an overrun error has occurred. Another character arrived before the system read the previously received character.
2	When set (1), a parity error has occurred. The parity of the incoming character did not match the programmed parity selection.
3	When set (1), a framing error has occurred. The first bit of the word and the stop bit handle framing. Either the system incorrectly received the transmitted character or the total number of characters in the received word did not match the programmed selection.
4	When set (1), remote BREAK key operation was detected. Some terminals have a BREAK key that, when pressed, will hold the data line low.
5	When set (1), the transmitter holding register is empty. The serial input/output channel is ready to receive another character for transmission.
6	When set (1), the transmitter shift register is empty. The specified serial input/output channel is not currently transmitting.
7	When set (1), a time-out error occurred. The receiving device did not respond within a reasonable period of time. A printer, when off line, will typically return this type of error.

Table 9-8. Modem Control Status

BIT	DESCRIPTION
0	When set (1), the clear to send (CTS) line has changed state.
1	When set (1), the data set ready (DSR) status has changed.
2	When set (1), the trailing edge of the ring signal has been detected.
3	When set (1), the carrier detect (CD) signal has changed state.
4	This bit reflects the current state (high or low) of the clear to send line.
5	This bit reflects the current state (high or low) of the data set ready line.
6	This bit reflects the current state (high or low) of the ring indicator line.
7	This bit reflects the current state (high or low) of the carrier detect line.

Input/Output Interrupts

Printer Input/Output (INT 17H)

INT 17H performs input/output functions for the parallel port on the computer. Before executing the interrupt, load the AH register with one of the function codes described in Table 9-9. This interrupt uses the parallel configuration table, set up by interrupt 18H, to complete its operation. Refer to "Parallel/Serial Configuration" in this chapter.

Table 9-9. Printer Input/Output Function Codes

FUNCTION CODE	DESCRIPTION
00H	Print the next character. Sends the character in the AL register to the port. If the parallel device does not return a ready status within a reasonable amount of time, set bit 0 of the AH register (1), otherwise the register will return the status report (refer to Table 9-10). Load the DX register with the port number to be used (00H for LPT1, 01H for LPT2, etc.).
01H	Initialize the printer port. Register AH will return the status report following execution of the interrupt (refer to Table 9-10).
02H	Read status of printer port. This function returns the status report (refer to Table 9-10) in register AH.

Table 9-10. Parallel Printer Status Report

BIT	DESCRIPTION
0	A timeout error has occurred. The parallel device is probably not ready, perhaps off line.
1	Not used.
2	Not used.
3	An input or output error has occurred.
4	The device is on line.
5	The out of paper signal is active.
6	The receiver acknowledges the transmitted character.
7	The device is busy or in an error state.

Input/Output Interrupts

Parallel/Serial Configuration (INT 18H)

INT 18H serves as a pointer to the parallel and serial channel device tables. These are the tables used by MS-DOS and many application packages to configure the computer's serial and parallel ports. This interrupt can also route the parallel printer output to a serial communications channel.

NOTE: In computers that contain BASIC in ROM, this interrupt points to the BASIC program. Programs that make a specific call to BASIC ROM routines will not work on the Advanced Desktop computer.

In order to maintain compatibility between various versions of MS-DOS, the Monitor program loads the default configuration tables. MS-DOS and application programs can either modify these tables or install new tables and modify the pointers accordingly. Figure 9-1 illustrates the format of the serial and parallel device parameter mapping table.

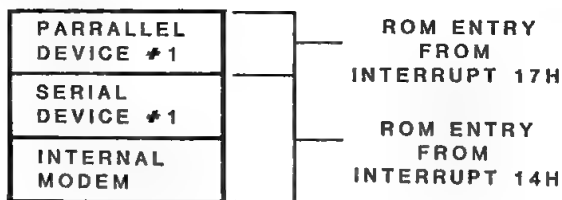


Figure 9-1. Serial and Parallel Device Layout

Input/Output Interrupts

Parallel Format

Table 9-11 shows the format for each of the three 4-byte parallel maps.

Table 9-11. Parallel Map Format

BYTE	PARAMETER DESCRIPTION
1	Performs remapping, parity, and uppercase/lowercase handling as described in Table 9-12.
2	Contains the ASCII value of the pad character used after a CR has been sent.
3	Contains the number of pad characters to send.
4	Contains the timeout value for the parallel device.

NOTE: You can modify these parameters by using the MS-DOS CONFIGUR utility supplied with Zenith Data Systems versions of MS-DOS for this computer.

Table 9-12. Parallel Map Byte #1

BIT	DESCRIPTION
0-3	Remapping. The value stored in these bits determines the parallel device mapping. If the value is 0, no mapping takes place. If the value is 1, the output is remapped to COM1. If the value is 2, the output is remapped to COM2 (Since the computer has only one COM port, do not use this option). Values of 3 or greater are not used.
4	If set (1), strip parity on output.
5	Not used.
6	If set (1), convert (map) lowercase output to uppercase.
7	Not used.

Input/Output Interrupts

Serial Format

Table 9-13 shows the format for the two 8-byte serial maps.

Table 9-13. Serial Map Format

BYTE	DESCRIPTION
1	Bit-mapped to indicate the type of handshaking that will be used by this channel (refer to Table 9-14).
2	Bit-mapped to indicate the attributes used by this channel for code transmission (refer to Table 9-15).
3	Contains the ASCII code for the pad character sent following a CR.
4	Contains the number of pad characters to send.
5	Used as a flag for burst transmission with ETX/ACK handshaking.
6	Used as a counter for ETX/ACK handshaking.
7	Bit-mapped to indicate the parameters used during initialization of this channel. (refer to Table 9-16).
8	Reserved.

Table 9-14, 9-15, and 9-16 contain the bit-mapped information for byte #1 (handshaking), #2 (attributes), and #7 (initialization parameters), respectively, of the serial map.

Table 9-14. Byte #1 (Handshaking)

BIT	DESCRIPTION
0	Compatibility bit. If set (1), use IBM-compatible DTR and RTS signals for handshaking when active-high. If clear (0), bit 1 determines the type of handshaking protocol used (hardware or software).
1	Hardware/software protocol bit. If set (1), use software handshaking; if clear (0), use hardware handshaking.
2	DC1/DC3, EXT/ACK, DTR handshaking, based on the condition of bit 1. With bit 1 set, a set condition (1) on this bit indicates DC1/DC3 protocol. If this bit is clear (0), use ETX/ACK protocol. With bit 1 clear, a set condition (1) on this bit indicates DTR handshaking protocol. If this bit is clear (0), DTR is not in use (see bit 4).

Input/Output Interrupts

-
- 3 DTR polarity bit, based on the condition of bits 1 and 2. If bit 1 is clear (0) (hardware handshaking is in effect) and bit 2 set (1) (DTR handshaking is used), this bit determines the polarity of DTR handshaking: if set (1), DTR handshaking is active high; if clear (0), DTR handshaking is active low.
If bit 1 is clear, software handshaking is in effect and this bit indicates if software is waiting for a handshake signal: if set (1), it is waiting; if clear (0), it is not.
- 4 RTS handshaking bit, based on the condition of bit 2. If bit 1 is clear (0), hardware handshaking is in effect. If this bit is set (1), RTS handshaking is used; if clear (0), RTS handshaking is not used.
- 5 RTS polarity bit, based on the condition of bits 1 and 4. If bit 1 is clear (0), hardware handshaking is in effect and this bit indicates the polarity of RTS handshaking when bit 4 is set (1). If this bit is set (1), RTS handshaking is active high; if clear (0), RTS handshaking is active low.
-

Table 9-15. Byte #2 (Parity and Case Mapping)

BIT	DESCRIPTION
0	If clear (0), do not strip the parity bit on input; if set (1), strip the parity bit on input.
1	If clear (0), do not strip the parity bit on output; if set (1), strip the parity bit on output.
2	If set (1), convert (map) lowercase alphabetic characters to uppercase on input; if clear (0), do not convert lowercase alphabetic characters to uppercase on input.
3	If set (1), convert (map) lowercase alphabetic characters to uppercase on output; if clear (0), do not convert lowercase alphabetic characters to uppercase on output.

Input/Output Interrupts

Table 9-16. Byte #7 (Word Length, Stop Bits, Parity, and Baud Rate)

BIT	DESCRIPTION
0-1	Determines the number of bits in a word. If the value is 0 (00), use five bits (not used in most PC-compatible applications). If the value is 1 (01), use six bits (not used in most PC-compatible applications). If the value is 2 (10), use seven bits. If the value is 3 (11), use eight bits.
2	Determines the number of stop bits. If set (1), use two stop bits. If clear (0), use 1 stop bit. Even though the hardware is capable of 1.5 stop bits, you cannot program this interrupt to use that value. You must program the hardware directly.
3-4	Determines the parity used. If the value is 0 (00) or 2 (10) no parity is used, a 1 (01) indicates odd parity, and a 3 (11) indicates even parity.
5-7	Determines the baud rate used (refer to Table 7-6).

Chapter 10

Disk Drive Interrupts

Chapter 6 provided general information about using and programming interrupts. This chapter provides specific information about the nine interrupts used for disk input, output, and setup. Table 10-1 lists the disk drive interrupts described in this chapter.

Table 10-1. Disk Drive Interrupts

INTERRUPT	FUNCTION
0EH	Floppy disk hardware interrupt return
13H	Floppy and hard disk software input/output call
19H	Operating system boot routine call
1EH	Floppy disk parameters
40H	Floppy disk software input/output call
41H	Hard disk 0 parameters
46H	Hard disk 1 parameters
4BH	Hard disk 2 parameters
76H	Hard disk hardware interrupt return

Floppy and Hard Disk Drive Hardware Interrupt Return (INT 0EH and INT 76H)

INT 0EH services command complete interrupts from the floppy disk controller (FDC). Upon completion of a command, the FDC returns an interrupt causing a jump to this vector. INT 76H performs the same function for the hard disk controller (HDC). These interrupt routines cannot be altered without affecting the performance of the Advanced Desktop computer. Do not modify these interrupts with user programs.

Disk Drive Interrupts

Disk Input/Output (INT 13H and INT 40H)

These interrupts provide access to the system's disk drives. They allow you to call a particular BIOS routine and carry out a disk operation such as reading or writing a sector. INT 13H is capable of handling either floppy disk or hard disk operations. INT 40H is capable of handling only floppy disk operations. When INT 13H is called, the routine checks the contents of the DL register. If the most-significant bit of the register is set (the drive identification code is 8X, indicating that a hard disk operation was requested) the request is handled. If the most-significant bit is clear (the drive identification code is 0X), the floppy disk operation is passed to the floppy disk driver routine and handled by INT 40H. The drive identification codes are listed in Table 10-3. Floppy operations may be passed directly to INT 40H without going through INT 13H.

Before calling a particular routine, the CPU registers must be initialized. Any interrupt that is called without first initializing the appropriate CPU registers will cause a jump to a return (RET) function (no operation). The discussion of each function provides specific information about initializing the registers. To identify a particular function, its code is loaded into the AH register. Table 10-2 describes the function codes used with INT 13H and INT 40H.

Table 10-2. Disk Drive Function Codes

CODE	FUNCTION	DESCRIPTION
00H	Reset disk system	Resets each disk drive and drive controller and reprograms the controllers for the appropriate drive characteristics.
01H	Read disk status	Returns the error status information for the drive last used in a disk input/output operation.
02H	Read sectors from a disk	Reads specified sector(s) into a buffer and returns error information.

Disk Drive Interrupts

Table 10-2 (continued). Disk Drive Function Codes

CODE	FUNCTION	DESCRIPTION
03H	Write sectors to a disk	Writes buffer contents to specified sector(s) and returns error information.
04H	Verify sector(s)	Tests specified sector(s) for readability and returns error information.
05H	Format a track	Formats specified tracks and returns error information.
08H	Return drive parameters	Returns drive parameter block information for the drive specified.
09H	Set hard drive parameters	Writes hard drive parameter block information from the tables pointed to by INT 41H and INT 46H to the HDC.
0AH	Read sectors with ECC bytes	Reads data and ECC bytes from specified sectors to specified memory block.
0BH	Write sectors with ECC bytes	Writes data and ECC bytes from specified memory block to specified number of sectors.
0CH	Seek	Moves the heads of the specified drive to the specified track or cylinder.
0DH	Reset hard disk drives	Resets the specified hard disk drive.
0FH	Return drive type	Returns drive type information for the drive specified.

Disk Drive Interrupts

Table 10-2 (continued). Disk Drive Function Codes

CODE	FUNCTION	DESCRIPTION
10H	Test	For hard disk drive operations, this function tests the status of the drive ready line. For floppy disk drive operations, it tests the disk changed line status.
11H	Set drive type/recalibrate	For floppy disk drive operations this function allows drive type to be specified prior to formatting. For hard disk drive operations, this function causes the specified drive to seek to Track 00.
12H	Set media type	Allows maximum number of heads, tracks, and sectors to be specified prior to formatting.
14H	HDC self-test	This function causes the hard disk controller (HDC) to execute its internal diagnostics and report any errors.
15H	Return DASD type	If a hard disk drive (Direct Access Storage Device) is installed, this function returns the number of 512 byte sectors available.
18H	Acknowledge cartridge change	This function call acknowledges the cartridge changed signal for the drive specified.
19H	Reinitialize DMA 360 drive	This function call causes the DMA 360 (removable cartridge) drive specified to reset. If the drive specified is not a DMA 360 drive, an error code is generated.

Disk Drive Interrupts

Table 10-2 (continued). Disk Drive Function Codes

CODE	FUNCTION	DESCRIPTION
1BH	Assert recovery mode	This function call is used locally by the hard disk drive interrupt handler in the event of errors during read operations on removable cartridge drives.
1CH	Deassert recovery mode	This function call is used locally by the hard disk drive interrupt handler in the event of errors during read operations on removable cartridge drives.

Drive Selection

The routines called by INT 13H or INT 40H control a maximum of two floppy disk drives and two hard disk drives. If your system contains more than two floppy disk drives or more than two hard disk drives, you will need to program the disk controllers directly. Refer to Chapter 16 for programming information. All function calls except reset drive system (00H) and read disk status (01H) require that you select a particular drive to perform the operation. To select a particular drive, load one of the drive codes listed in Table 10-3 into the DL register.

Table 10-3. Drive Identification Codes

CODE	DRIVE
00H	Floppy disk drive 1
01H	Floppy disk drive 2
80H	Hard disk drive 1
81H	Hard disk drive 2

Disk Drive Interrupts

Error Status Codes

Many of the function calls return an error status code in either the AH or the AL register. The description of the individual functions indicates which register to read for that particular call. Table 10-4 lists each of the codes and describes the disk status indicated.

Hard disk drive operations retry four times before setting the CF flag and sending the error status code to either the AH or the AL register. If either register or the carry flag indicates an error, reset the drive using function code 00H and then retry the operation.

Floppy disk drive operations do not retry. The carry flag is set and an error status code is sent to the appropriate register as soon as an error is encountered. If the CF flag and AH or AL register indicate an error occurred, reset the drive using function code 00H and retry the operation at least three times.

Table 10-4. INT 13H and INT 40H Error Status Codes

CODE	DISK STATUS OR ERROR
00H	No error. The last operation was successfully completed.
01H	Illegal function. An illegal function number was called.
02H	No address mark. The address mark or sector header was not found.
03H	Write protected. A write operation was attempted to a write-protected disk.
04H	Sector not found. The selected sector could not be located.
05H	Reset failed. The reset operation failed. Track 00 was not detected or invalid drive information was stored in CMOS.
06H	Disk change. The disk has been changed since the last disk I/O operation.
07H	Parameter failure. The drive parameters could not be sent to the controller.

Disk Drive Interrupts

Table 10-4 (continued). INT 13H and INT 40H Error Status Codes

CODE	DISK STATUS OR ERROR
08H	DMA overrun. A DMA overrun occurred and data was lost.
09H	Transfer incomplete. The data to be transferred would not fit in the specified segment.
0BH	Bad track. The controller detected a bad track flag.
10H	CRC error. The CRC read from the disk did not match the calculated CRC value. (Refer to Chapter 16 for the CRC polynomial.)
11H	Soft error. The controller detected a correctable error and the data read was corrected using ECC.
20H	Invalid information. The FDC or HDC responded to the function call with invalid information.
40H	Seek failure. The drive could not seek to the specified track.
80H	No response. The FDC or HDC did not respond to the function call.
BBH	Undefined error. The controller responded with an undefined error code.
FFH	No drive status. The sense drive status operation failed.

Function Call Codes

Function Code 00H: Reset Disk System — Calls the interrupt routine that resets each disk drive in the system. The reset sequence moves the read/write head(s) to track 0. This interrupt also resets both drive controllers and programs them with the drive characteristics stored in the drive parameter tables pointed to by INT 1EH, INT 41H, INT 46H, and INT 4BH.

Disk Drive Interrupts

Function Code 01H: Read Disk Status — Calls the interrupt routine that returns the disk status code in register AL. Use this call when an applications program needs to see the disk status generated by the last disk operation. Table 10-4 describes the error status codes returned by this operation.

Function Code 02H: Read Sector(s) — Calls the interrupt routine that reads the specified number of sectors into the disk buffer. Table 10-5 lists the information that must be loaded into each CPU register prior to calling this function. This information is also used to initialize the CPU for write sectors, verify data, read sectors and ECC bytes, and write sectors and ECC bytes function calls.

Table 10-5. CPU Register Initialization — Function Codes 02H-04H and Function Codes 0AH-0BH

REGISTER	CONTENTS
AH	Initialize this register with the function code.
AL	Initialize this register with the number of sectors to be transferred.
CH	For floppy drive operations, initialize this register with the starting track. For hard disk operations, initialize this register with the eight least-significant bits of the starting cylinder number.
CL	For floppy drive operations, initialize this register with the logical starting sector number. For hard drive operations, initialize the six most-significant bits with the logical starting sector number and the two least-significant bits with the two most-significant bits of the starting cylinder number.
DH	Initialize this register the head or disk side number. Use either the value 00H or 01H if register DL specifies a floppy disk drive. Use a value between 00H-0FH if DL specifies a hard disk drive.
DL	Refer to Table 10-3 and initialize this register with the drive identification code.

Disk Drive Interrupts

Table 10-5 (continued). CPU Register Initialization — Function Codes 02H–04H and Function Codes 0AH–0BH

REGISTER	CONTENTS
ES:BX	For function call 02H, initialize this register pair with the pointer to (address of) the disk buffer the segment number in register ES and the offset address in register BX. For function call 03H, these registers will contain a pointer to the memory address for the data to be transferred. The ES:BX register pair need not be initialized for function call 04H since the call verifies data without actual transferring it to the CPU.

Upon completion of the operation, the controller returns drive status information through the AH register and the full carry (CF) flag. If the operation was successful, the CF flag will be clear. The AH register will contain one of the error status codes listed in Table 10-4.

Function Code 03H: Write Sector(s) — Calls the interrupt routine that writes the contents of the disk buffer into the specified number of sectors on the disk. Refer to Table 10-5 for the initialization information required for this operation. As with the read sector call, upon completion of the operation, the drive controller returns drive status information through the AH register and CF flag. The AH register will contain the error status code (refer to Table 10-4) and the CF flag will be cleared if the operation was successfully completed.

Function Code 04H: Verify Sector(s) — Calls the interrupt routine that tests the specified sector(s) to verify that the data is readable. If errors that cannot be corrected using either ECC or CRC methods are detected, the CF flag is set. If correctable errors are encountered, the operation is considered successfully completed and the CF flag is cleared. In either case, an error code (refer to Table 10-4) is returned in the AH register. The data read is not transferred during this procedure. Table 10-5 lists the initialization information required for this operation.

Disk Drive Interrupts

Function Code 05H: Format a Track — Calls the interrupt routine that formats the specified track. You must place the sector header information for each sector on the track in a block in memory. Table 10-6 lists the initialization information that must be loaded into the CPU registers prior to calling this function. The CF flag will be set upon completion of the operation if an error occurred. The error status code (see Table 10-4) will be returned in register AH.

Table 10-6. CPU Register Initialization — Function Code 05H

REGISTER	CONTENTS
AH	Initialize this register with the function code.
AL	For floppy drive operations, initialize this register with the number of sectors per track. For hard drive operations, initialize this register with the value (01H to 10H) of the interleave. For example, for an interleave of 2 to 1, enter 02H.
CH	For hard drive operations, initialize this register with the eight least-significant bits of the logical starting cylinder. This register is not used for floppy drive operations.
CL	For hard drive operations, initialize two least-significant bits of this register with the two most-significant bits of the logical starting cylinder number. This register is not used for floppy drive operations.
DH	For floppy drive operations, initialize this register with the side select number (00H or 01H). For hard drive operations, initialize this register with the head number (00H–0FH).
DL	Refer to Table 10-3 and initialize this register with the drive identification code.
ES:BX	For floppy drive operations, this register pair is initialized as a pointer to a memory block containing the sector header information for each of the sectors to be formatted. The register pair is not used for hard drive operations.

Disk Drive Interrupts

Function Code 08H: Return Drive Parameters — Calls the interrupt routine that reads and reports the drive parameters block for the drive specified in the DL register. To execute the function call, load the function code into register AH and the drive identification code from Table 10-3 into the DL register. The controller then reads the parameters block and returns the drive parameters information to one of the CPU registers. Table 10-7 lists the CPU registers and their contents. For floppy drive operations, if no drives are installed, each of the registers listed will contain 00H. If the drive number is invalid, the drive type is unknown, or CMOS drive information is invalid, bad, or not present, the value returned in the DL register will be valid and all other registers will contain 00H. For hard drive operations, if an error occurs an error status code (see Table 10-4) will be returned in register AH. In either case, if an error occurs the CF flag will be set.

Table 10-7. CPU Register Results — Function Code 08H

REGISTER	CONTENTS
AH	This register returns the value 00H for all floppy operations. For hard drive operations, it returns an error status code.
BL	For floppy drive operations, this register returns the drive type information stored in CMOS. It is not used for hard drive operations.
CH	For floppy drive operations, this register returns the last addressable track number. For hard drive operations, the last addressable cylinder number is returned.
CL	This register returns the last addressable sector number.
DL	This register returns the number of drives installed.
DH	This register returns the last addressable head number for the drive selected.
ES:DI	For floppy drive operations, this register pair returns an address pointer for the drive parameters block for the selected drive. The register pair is not used for hard drive operations.

Disk Drive Interrupts

Function Code 09H: Set Hard Drive Parameters — Calls the interrupt routine that causes the drive parameters block pointed to by INT 41H to be sent to the controller as drive characteristics for hard disk drive 0 and the drive parameters block pointed to by INT 46H to be sent as drive characteristics for hard disk drive 1. The format for the drive parameters blocks is described Table 10-16. To execute the function call, load the function code into the AH register and the starting drive code (see Table 10-3) into the DL register. If the drive characteristics cannot be transferred to the controller, the CF flag will be set and the AH register will return an error status code.

Function Code 0AH: Read Sectors with ECC Bytes — Calls the interrupt routine that reads the specified long sectors into a specified memory block. This function is similar to the read sectors function (function code 02H) but is valid for hard drive operations only. Typically, it causes 516-byte sectors (512 data bytes and 4 ECC bytes) to be read from hard disks. Table 10-5 lists the CPU register initialization information required for this function call. Upon completion of the operation, an error status code (see Table 10-4) is returned to register AH. If the operation was successfully completed, the CF flag will be cleared.

Function Code 0BH: Write Sectors with ECC Bytes — Calls the interrupt routine that writes long sectors from a specified memory block into specified sectors. This function is similar to the write sectors function (function code 03H) but is valid only for hard drive operations. Typically, it causes 516-byte sectors (512 data bytes and 4 ECC bytes) to be written. Table 10-5 lists the CPU register initialization information required for this function call. Upon completion of the operation, an error status code (see Table 10-4) is returned to register AH. If the operation was successfully completed, the CF flag will be cleared.

Disk Drive Interrupts

Function Code 0CH: Seek — Calls the interrupt routine that moves the heads of the specified hard disk drive to the specified cylinder. Table 10-8 lists the CPU initialization information required for this function call. Upon successful completion of the operation the CF flag is cleared. The error status code (see Table 10-4) is returned in register AH.

Table 10-8. CPU Register Initialization — Function Code 0CH

REGISTER	CONTENTS
AH	Initialize this register with the function code.
CH	Initialize this register with the eight least-significant bits of the cylinder number.
CL	Initialize the two least-significant bits of this register with the two most-significant bits of the cylinder number.
DL	Initialize this register with a hard disk drive identification code from Table 10-3.

Function Code 0DH: Reset Hard Disk Drive — Calls the interrupt routine that resets the specified hard drive. This function call is similar to the reset disk system function (function call 00H) but resets only the specified hard drive. To execute this call, initialize the DL register with a hard disk drive identification code from Table 10-3. If the operation is successfully completed the CF flag will be cleared. An error status code (see Table 10-4) is returned in register AH. This command is valid for hard drive operations only.

Function Code 0FH: Return Drive Type — Calls the interrupt routine that reads and reports the drive type information for the specified drive. To execute this call, initialize the AH register with the function code and the DL register with a drive identification code from Table 10-3. Upon completion of the command, a device type code will be returned in the AL register. Table 10-9 describes the device type codes.

Disk Drive Interrupts

Table 10-9. CPU Register Results — Function Code 0FH

DEVICE TYPE CODE	DESCRIPTION
00H	This result byte indicates a non-existent device.
01H	This result byte indicates that the drive is a removable media device and no media change was detected.
02H	This result byte indicates that the drive is a removable media device and a media change was detected.
03H	This result byte indicates that the drive is a fixed media device.

Function Code 10H: Test — Calls the interrupt routine to check the status of the drive ready line for the specified hard disk drive or the status of the media changed line for floppy disk drives. To execute this call, initialize the AH register with the function code and the DL register with a disk drive identification code from Table 10-3.

Upon completion of this operation on floppy drives, the CF flag will be set if a media change has been detected, or clear if no media change is detected. Table 10-10 describes the media state code that will be returned in the AH register. Upon completion of the operation on a hard drive, the CF flag will be set if the drive is not ready, or cleared if the drive is ready. An error status code (see Table 10-4) will be returned to the AH register.

Table 10-10. CPU Register Results — Function Code 10

MEDIA STATE CODE	DESCRIPTION
00H	This code indicates that the media changed line was inactive.
06H	This code indicates that the media changed line was active.
80H	This code indicates that the operation was attempted on a drive that was not present in the system.

Disk Drive Interrupts

Function Code 11H: Set Drive Type /Recalibrate — For hard drive operations, this function calls the interrupt routine that recalibrates the specified drive (move the read/write heads to cylinder 0). To execute this call, register AH is initialized with the function code and register DL is initialized with the drive identification code from Table 10-3. Upon completion of the operation, the CF flag is set if an error was encountered. An error status code (see Table 10-4) is returned in the AH register.

For floppy drive operations this function calls the interrupt routine that sets the device type for the specified floppy drive. Either this function call or function call 12H must be successfully executed prior to formatting a floppy disk. To execute this function call, initialize the AH register with the function code, the DL register with a drive identification code from Table 10-3, and the AL register with a device type code from Table 10-11. Upon completion of the operation, an error status code (see Table 10-4) will be returned in register AH.

Table 10-11. CPU Register Initialization — Function Code 11H

DEVICE

TYPE

CODE	DESCRIPTION
------	-------------

01H	This code indicates a 360K disk in a 360K drive.
-----	--

02H	This code indicates a 360K disk in a 1.2M drive.
-----	--

03H	This code indicates a 1.2M disk in a 1.2M drive.
-----	--

04H	This code indicates a 720K disk in a 1.2M drive.
-----	--

Function Code 12H: Set Media Type — Calls the interrupt routine that initializes the FDC with the last addressable track and sector number prior to formatting a disk. Either this function call or function call 11H must be successfully completed before a formatting operation can be executed. To execute this call, initialize the AH register with the function code, the DL register with a floppy drive identification code (see Table 10-3), the CH register with the last addressable track number, and the CL register with the last addressable sector

Disk Drive Interrupts

number. Upon successful completion of the operation, the CF flag will be cleared. Table 10-12 lists the result codes returned to the AH register. This function is valid for floppy drive operations only.

Table 10-12. CPU Register Results — Function Code 12H

RESULT CODE	DESCRIPTION
00	This code indicates that the operation was successfully completed and the input parameters are valid.
01	This code indicates that the operation was not completed and that the function is not supported.
0C	This code indicates that the operation was not completed and that the input parameters are invalid.

Function Code 14H: Hard Disk Controller Self-Test — Calls the interrupt routine that runs the controller's self-tests. These tests include a check of the controller's processor, data buffer, ECC circuits, and ROM checksum. To execute these tests, initialize the AH register with the function code. Upon successful completion of the tests, the CF flag will be cleared. An error status code (see Table 10-4) is returned in the AH register. This function is valid for hard disk drive operations only.

Function Code 15H: Return DASD (Direct Access Storage Device) Type — Calls the interrupt routine that returns the number of 512 byte data sectors available on the selected hard disk drive. This function code is valid for hard drive operations only. To execute this call, initialize the AH register with the function code and the DL register with a hard drive identification code. The routine returns 00H in the AH register if the operation was attempted on a drive not present in the system, 03H if the drive specified was a valid hard disk drive. The CX and DX registers return the number of available data sectors. This number is valid only if the AH register contains the 03H valid drive code.

Function Code 18H: Acknowledge Cartridge Changed — Calls the interrupt routine that acknowledges and deasserts the cartridge changed signal. To execute this call, initialize the AH register with the function code and the DL register with a hard drive identification code from Table 10-3. Upon successful completion of the operation, the CF flag is cleared. An error status code (see Table 10-4) is returned in the AH register. This function is valid for removable cartridge hard drives only.

Function Code 19H: Reinitialize DMA 360 (Removable Cartridge) Drive — Calls the interrupt routine that reinitializes the specified removable cartridge hard disk drive. To execute this function call, load the function code into register AH and a hard disk drive identification code from Table 10-3 into the DL register. If the operation is attempted on a drive that is not a DMA 360 drive or if an error occurs, the CF flag is set. An error status code (see Table 10-4) is returned in the AH register. This function is valid for removable cartridge hard disk drives only.

Function Code 1BH: Assert Recovery Mode — Calls the interrupt routine that asserts the recovery mode line on removable cartridge drives. This routine is used locally by the hard disk interrupt handler when an error is detected during a read operation. This function is valid for removable cartridge hard drives only.

Function Code 1CH: Deassert Recovery Mode — Calls the interrupt routine that deasserts the recovery mode line on removable cartridge drives. This routine is used locally by the hard disk interrupt handler when an error is detected during a read operation. This function is valid for removable cartridge hard drives only.

Booting an Operating System (INT 19H)

INT 19H attempts to read from track 0, sector 1 of floppy drive 0. The code located at this position is loaded into memory and executed. If the operation is unsuccessful, the routine then attempts to boot hard drive 0. If the boot routine fails again, an error message will appear and control passes to the Monitor program.

Floppy Disk Parameters (INT 1EH)

INT 1EH contains an address pointer. The pointer allows access to an area in ROM that contains the disk parameter tables for the supported floppy disk drive and media combinations. When you turn the computer on, the pointer is loaded with the starting address for the appropriate disk parameter table. Table 10-13 describes the structure of the tables.

Table 10-13. ROM Disk Parameters Table

BYTE	FUNCTION	DESCRIPTION
01	RSpecify 1	Bits 0–3 contain the head unload time (the range is 16–240 milliseconds in 16-millisecond increments). Bits 4–7 contain the head step rate for the drive (the range is 1–16 milliseconds in 1-millisecond increments, in reverse order — FH = 1 millisecond, EH = 2 milliseconds, and so on). Typically, this byte will contain 0FH.
02	RSpecify 2	Bits 1–7 contain the head load time (the range is 2–254 milliseconds in 2-millisecond increments). Bit 0 contains the DMA flag. If bit 0 is clear (0), the DMA mode is in use. This byte will contain 02H.
03	RMotor delay	This is the amount of time the motor will remain on after the last disk operation, measured in timer ticks. The timer tick rate is 18.2 timer ticks per second, or one every 55 ms. A value of 25H is used for all supported media and drive combinations.
04	RBytes per sector	This contains a hex code that sets the number of data bytes per sector. A value of 02H, indicating 512 bytes per sector, is used for all supported drive and media combinations.

Disk Drive Interrupts

Table 10-13 (continued). ROM Disk Parameters Table

BYTE	FUNCTION	DESCRIPTION
05	RSectors per track	This contains the number of the last addressable sector on the track. 09H indicates a double-density disk. 0FH indicates a high-density disk.
06	RGap length	This contains the hex code that establishes the length of gap 3 for read, write, and verify operations. Refer to Table 10-14. The information contained in this byte overrides any user programmed gap length during INT 13H and INT 40H operations.
07	RData length	This byte allows partial sectors. If the bytes per sector code equals 00H, then any number between 00H and FFH may be entered, allowing a sector length from 1 byte to 255 bytes. Enter FFH if the bytes per sector code (byte 4) is a number other than 00H.
08	Reformat gap length	This byte contains a hex code that sets the length of gap 3 for format operations. This code is overridden by the format gap length contained in any user-programmed disk parameter tables. See Table 10-14.
09	Reformat data	This byte contains the bit pattern used by the format command to fill and test the sector. Typically, the bit pattern 1110 0101 (E5H) is used.

Disk Drive Interrupts

Table 10-13 (continued). ROM Disk Parameters Table

BYTE	FUNCTION	DESCRIPTION
0A	RHead settle	This contains the head settle time, measured in 1-millisecond increments. A value of 0FH is used for all supported drive and media combinations. ¹
0B	RMotor start	This contains the motor start time, measured in 125-millisecond increments. A value of 08H is used for all supported drive and media combinations. ²
0C	RCylinders per disk	This byte contains the number of the last addressable track on the disk. 4FH indicates a high-density (80 track) disk. 27H indicates a double-density (40 track) disk.
0DH	RData transfer rate	This byte contains a hex code that indicates the data transfer rate. 00H indicates a transfer rate of 500kbps, 40H indicates 300kbps, and 80H indicates 250kbps.

NOTES

1. A minimum head settle time of 15 mS for high-density drives and 20 ms for double-density drives is enforced for write operations.
2. A minimum motor start time of 1 second is enforced for write and format operations. A minimum motor start time of 625 ms is enforced for read or verify operations.

Disk Drive Interrupts

The Advanced Desktop computer supports the following six media/drive combinations with ROM disk parameter tables:

- 360K media in a 360K drive (360/360)
- 360K media in a 1.2M drive (360/1.2)
- 1.2M media in a 1.2M drive (1.2/1.2)
- 720K media in a 720K drive (720/720)
- 720K media in a 1.44M drive (720/1.44)
- 1.44M media in a 1.44M drive (1.44/1.44)

Table 10-14 lists the bytes per sector, gap length, and format gap length parameters used for each combination.

Table 10-14. Drive Parameters — INT 1EH

MEDIA/ DRIVE	BYTES PER SECTOR (DECIMAL)	BYTES PER SECTOR CODE	SECTORS PER TRACK	GAP LENGTH	FORMAT GAP LENGTH
360/360	512	02H	09H	2AH	50H
360/1.2	512	02H	09H	2AH	50H
1.2/1.2	512	02H	0FH	1BH	54H
720/720	512	02H	09H	2AH	50H
720/1.44	512	02H	09H	2AH	50H
720/1.44	512	02H	09H	2AH	50H
1.44/1.44	512	02H	12H	1BH	6CH

Disk Drive Interrupts

User-programmed disk parameter tables may be created for this computer. These tables must be created if you intend your software to be used on older PC-compatible systems. Many older systems do not have built-in ROM disk parameters and rely entirely on user-programmed disk parameter tables. Tables 10-15 and 10-16 contain the information for building your own drive parameters table. In systems like this computer, the ROM disk parameter table overrides some of the information contained in user-programmed disk parameter tables as noted in the tables. Also, the user-programmed disk parameter table does not permit a variable last addressable track parameter or a variable data transfer rate. Use the appropriate ROM disk table where these parameters are required.

Table 10-15. Floppy Disk Parameter Table

BYTE	FUNCTION	DESCRIPTION
01	Specify 1	Bits 0–3 contain the head unload time (the range is 16–240 milliseconds in 16-millisecond increments). Bits 4–7 contain the head step rate for the drive (the range is 1–16 milliseconds in 1-millisecond increments, in reverse order — 0FH = 1 millisecond, 0EH = 2 milliseconds, and so on).
02	Specify 2	Bits 1–7 contain the head load time (the range is 2–254 milliseconds in 2-millisecond increments). Bit 0 contains the DMA flag. If bit 0 is clear (0), the DMA mode is enabled.
03	Motor delay	This is the amount of time the motor will remain on after the last disk operation, measured in timer ticks. The timer tick rate is 18.2 timer ticks per second, or one every 55 ms.
04	Bytes per sector	This contains the number of data bytes per sector. Refer to Table 10-13.
05	Sectors per track	This contains the number of sectors per track.

Disk Drive Interrupts

Table 10-15 (continued). Floppy Disk Parameter Table

BYTE	FUNCTION	DESCRIPTION
06	Gap length	This contains the gap length of gap 3 for read and write operations. Refer to Table 10-14. This information is overridden by the gap length specified in the most similar ROM disk parameters table during INT 13H and INT 40H operations.
07	Data length	If the bytes per sector parameter equals 00H, this byte specifies bytes per sector. Any value between 00H and FFH may be entered. This allows partial sector read, write, and verify operations. If the bytes per sector parameter (byte 04) is not 00H, enter FFH.
08	Formats gap length	This contains the gap length of gap 3 for format operations. Refer to Table 10-13. "Reformat gap length" does not override this parameter.
09	Format data	This byte contains the bit pattern used by the format command to fill and test the sector. Typically, the bit pattern 1110 0101 (E5H) is used.
0A	Head settle time	This contains the head settle time, measured in 1-millisecond increments. ¹
0B	Motor start time	This contains the motor start time, measured in 125-millisecond increments. ²

NOTES

1. A minimum head settle time of 15 ms for high density drives and 20 ms for double-density drives is enforced for write operations.
2. A minimum motor start time of 1 second is enforced for write and format operations. A minimum motor start time of 625 ms is enforced for read or verify operations.

Disk Drive Interrupts

Hard Disk Parameters (INT 41H, INT 46H, and INT 4BH)

These interrupts contain address pointers to memory blocks that contain the hard disk drive parameter tables. INT 41H contains the vector for hard drive 0, INT 46H for hard drive 1, and INT 4BH for hard drive 2. Table 10-16 describes the hard disk parameter table structure.

Table 10-16. Hard Disk Parameter Table

ENTRY	FUNCTION	DESCRIPTION
01	Max cylinders	This 16-bit data word contains the last addressable cylinder number.
02	Max heads	This 8-bit data byte contains the last addressable head number.
03	Not used	Fill this 16-bit data word with 0000H.
04	Precomp	This 16-bit data word contains the number of the first precompensated cylinder. 0000H indicates that precompensation begins on the first cylinder. FFFFH indicates that no precompensation is required.
05	ECC length	This 8-bit data word contains the maximum number of error bytes that can be corrected with ECC.
06	Options 200	Enter 08H if the drive has more than eight addressable heads.
07-09	Not used	Fill these 8-bit data bytes with 00H.
0A	Landing zone	This 16-bit data word contains the number of the cylinder to be used as a head landing zone.

Disk Drive Interrupts

Table 10-16 (continued). Hard Disk Parameter Table

ENTRY	FUNCTION	DESCRIPTION
0B	Sectors per track	This 8-bit data byte contains the number of the last addressable sector on the track. This computer accepts only 17 sectors per track formats for hard disks. Enter 11H.
0C	Reserve 200	This 8-bit data byte contains a hex code that indicates whether the drive is a removable cartridge type, servo-stepper type, or removable cartridge and servo-stepper type. Enter 01H for cartridge types, 02H for servo-stepper types, and 03 for drives that are removable cartridge and servo-stepper types.



Chapter 11

Video Interrupts

This chapter describes the video interrupts and how these interrupts are used in programming the video system in your computer. By using a Z-449 31 kHz video card, this computer is capable of operating in color graphics mode (CGA), enhanced graphics mode (EGA), Hercules mode, and monochrome video mode (MDA). Operating characteristics and functional abilities are different for each of these video modes.

Programming the Video Interrupts

Table 11-1 lists the video interrupts discussed in this chapter. Refer to Chapter 6 for information on programming with the system interrupts and to Chapter 17 for detailed information on the registers of the video controller and their use.

Table 11-1. Video Interrupts

INTERRUPT	FUNCTION
10H	Video input/output
1DH	Video initialization
1FH	Defining characters

Video Input/Output (INT 10H)

INT 10H provides communications with the video logic section of the computer. To use this interrupt, place one of the function codes described in Table 11-2 in register AH. The following pages contain descriptions of each of the function codes.

Video Interrupts

Table 11-2. Video Input/Output Function Codes

CODE	FUNCTION
00H	Set video mode
01H	Set cursor type
02H	Set cursor position
03H	Read cursor position
04H	Read light pen position (not used)
05H	Select active display page
06H	Scroll an area of screen up
07H	Scroll an area of screen down
08H	Read cursor position
09H	Send character/attribute to screen
0AH	Send character to screen
0BH	Set graphics foreground color
0CH	Write graphics pixel
0DH	Read graphics pixel
0EH	Dumb terminal display
0FH	Return video state
10H	Set palette registers
11H	Character generator routine
12H	Alternate select
13H	Write string
64H	Set scrolling mode

Function Code 00H: Set Video Mode — Causes the interrupt to set the video mode according to the value placed in register AL and described in Table 11-3.

Table 11-3. Video Modes

MODE	DESCRIPTION
00H	40 characters by 25 rows, monochrome text display at the RGB output.
01H	40 characters by 25 rows, color text display at the RGB output.
02H	80 characters by 25 rows, monochrome display at the RGB output. Individual video pages may be scrolled without affecting other video pages.
03H	80 characters by 25 rows, color text display at the RGB output. Individual text pages may be scrolled.

Table 11-3 (continued). Video Modes

MODE	DESCRIPTION
04H	320 × 200 pixel resolution, color graphics display at the RGB output. Text displays appear as 40 characters by 25 rows.
05H	320 × 200 pixel resolution, monochrome graphics display at the RGB output. Text displays appear as 40 characters by 25 rows.
06H	640 × 200 pixel resolution, monochrome graphics display at the RGB output. All three scrolling modes are available. Text displays appear as 80 characters by 25 rows.
07H	80 characters by 25 rows, monochrome text display with a monochrome display adapter card.
0DH	320 × 200 pixel resolution, color graphics at the RGB output. Text displays appear as 40 characters by 25 rows.
0EH	640 × 200 pixel resolution, color graphics at the RGB output. Text displays appear as 80 characters by 25 rows.
0FH	640 × 350 pixel resolution, graphics display using a high-resolution monochrome monitor.
10H	640 × 350 pixel resolution, color graphics display using a high-resolution color monitor.
12H	640 × 480 pixel resolution, color graphics display using a 31 kHz analog color monitor. 16 colors available from a palette of 256,000.

NOTE: The 31 kHz video card supplied with this computer supports a wide range of video modes. These modes supersede the video modes described above. Refer to the "EGA Video Considerations" section later in this chapter for modes available with this card.

Function Code 01H: Set Cursor Type — Causes the interrupt to set the cursor size. Load bits 0 – 4 of the CH register with the starting scan line number of the cursor and bits 0 – 4 of the CL register with the ending scan line number. For a full block cursor, the starting scan line bits must equal 0 and the ending scan line bits must equal 7. While it is possible to load any value between 0 and 31, use only values 0 – 7 for the starting and ending scan line.

NOTE: The starting scan value may not be greater than the ending scan value, or no cursor will appear.

Video Interrupts

Function Code 02H: Set Cursor Position — Causes the interrupt to place the cursor at the specified row and column location on the screen. Load register DH with the cursor row number and register DL with the column number. The row numbers extend from 0 to 24. The column numbers extend from 0 to 79 (80 characters per line) or 39 (40 characters per line). Therefore, when you load a value of 0 in the DH and DL registers, the cursor appears in the upper left-hand corner of the screen. Load register BH with the video page number; this must be consistent with the video mode selected (only page 0 is valid when you are in the graphics mode).

Function Code 03H: Read Cursor Position — Causes the interrupt to return the current cursor information. The row number will be in register DH, the column number will be in register DL. The cursor's starting scan line number will be in register CH, and the cursor's ending scan line number will be in register CL. Before executing the interrupt, register BH must contain the page number.

Function Code 04H: Read Light Pen Position — A light pen option is not currently supported with this computer, even though this is a valid function code for PC-compatible computers.

The function code causes the interrupt to attempt to obtain the light pen's position. After the interrupt has been executed, register AH will contain the light pen trigger/switch status (0 or 1). A value of 1 in the AH register indicates an active switch closure (the light pen is on). If the register contains a 0 instead, the switch is not activated (the light pen is off). If the AH register contains a 1, then a number of other registers contain related position information. The DH register will contain the row number, register DL will contain the column number. The CH register will contain the scan line row number (0 – 199), and register BX will contain the pixel column number. The pixel column number can vary between 0 – 319 or 0 – 639, depending upon the graphics mode.

Function Code 05H: Select Active Display Page — Causes the interrupt to display the page specified in register AL. In the text modes, unused portions of video memory can serve as additional video pages. In video modes 0 and 1, the valid page numbers are 0 to 7; in video modes 2 and 3, they are 0 – 3. In the graphics modes, only page 0 is valid.

Function Code 06H: Scroll an Area of the Screen Up — Causes the interrupt to scroll the specified area of the screen up a specified number of lines. Prior to executing this interrupt, you must load certain registers with specific data. The CX register must contain the upper left-hand coordinates (place the row number in register CH and the column number in register CL) of the scroll area. The DX register must contain the lower right-hand coordinates (place the row number in register DH and the column number in register DL) of the same area. The BH register must contain the attribute byte for the blank lines, and the AL register contains the number of lines to scroll. If the AL register contains a 0, the entire window will be cleared.

NOTE: If a hardware or software smooth scrolling mode is active when this interrupt executes, it will function only if the entire screen is scrolled. Hardware and software scrolling operations affect the entire screen and not just a portion of the screen.

Function Code 07H: Scroll an Area of the Screen Down — Causes the interrupt to scroll the specified area of the screen down the specified number of lines. Before executing this interrupt, you must load the following register information. The CX register must contain the upper left-hand coordinates (place the row number in register CH and the column number in register CL) of the scroll area. The DX register must contain the lower right-hand coordinates (place the row number in register DH and the column number in register DL) for the same area. The BH register must contain the attribute byte for the blank lines, and the AL register must contain the number of lines to scroll. If the AL register contains a 0, the entire window will be cleared.

Function Code 08H: Read Cursor Position Contents — Causes the interrupt to return the character and attribute codes for the character that resides at the current cursor position. In text modes, register BH must contain the video page number. Upon completion of the routine, the character code will be in register AL and the attribute code will be in register AH.

Function Code 09H: Send Character and Attribute to Screen — Causes the interrupt to write the specified character and attribute codes to the cursor location. Place the character code in register

Video Interrupts

AL, and the attribute code in the BL register. The CX register contains the number of times the character is to repeat and, in text modes, the page number is in the BX register.

Function Code 0AH: Send Character to Screen — Causes the interrupt to write the specified character to the screen, but not the attribute byte. Place the character code in register AL, the number of times to repeat the character in register CX and, in text modes, the page number in register BH.

Function Code 0BH: Set Graphics Foreground Color — This function code is only available in mode 4, the 320 × 200 graphics mode. Place a value from 0 to 127 in register BH (see the following text) and a value from 0 to 4 in register BL.

If the value placed in register BH is even, the current background color will become the foreground color (normally, 0 – 31). Values above 15 will select the intensified level of the 16 colors.

If the value placed in register BH is odd, the value in the BL register will determine which one of two available palettes is chosen. The palette and pixel color number will determine the foreground color. A value of 0 will select palette 0 and a value of 1 will select palette 1. Refer to Table 11-4 for the palette number and color number matrix.

Table 11-4. Palette and Pixel Colors

COLOR NUMBER	PALETTE 0	PALETTE 1
1	Green	Cyan
2	Red	Magenta
3	Yellow	White

Function Code 0CH: Write Graphics Pixel — Causes the interrupt to light a single pixel at the specified location on the screen. Place the pixel row number in register DX, the pixel column number in register CX, and the color in the AL register. Row numbering extends from 0 – 199 and column numbering from 0 – 319 or 0 – 639.

Video Interrupts

Color values range from 0 – 3 in medium-resolution (320 × 200) mode or 0 – 1 in high-resolution (640 × 200). In all cases, 0 is the background color (usually black). In the high-resolution mode, 1 is the foreground color. In the medium-resolution mode, the palette and the color number determine the color, as described in Table 11-4.

If the most-significant bit (bit 7) of AL is set, the color is XORed with the current color, permitting simple animation. For more information on animation techniques, see the discussion on graphics in the GW-BASIC manual.

Function Code 0DH: Read Graphics Pixel — Returns the color of the pixel at the specified location. Place the pixel row number in register DX, and the pixel column number in register CX. The returned color value will appear in the AL register.

Function Code 0EH: Dumb Terminal Display — Causes the interrupt to treat the character as if it were sent to a dumb terminal. The routine will treat the back space (08H), carriage return (0DH), line feed (0AH), and bell (07H) as console commands rather than screen formatting characters. If a character should print at the end of a screen line, the cursor will position at the start of the next line. If you perform a line feed on the last display line of the screen or if a character prints at the last position of the last line, the screen will scroll up one line. When scrolling, the attribute for the new row (when in text mode) will be the same as the attribute of the character at the cursor position on the line when the scrolling takes place.

Place the character in register AL, the foreground color in register BL (for graphics modes), and the display page number in register BH.

Function Code 0FH: Return Video State — Returns the current video state. Register AL will contain the current video mode (refer to Function Code 00H). The AH register will contain the screen width in columns, and register BH will contain the active video page number.

Video Interrupts

Function Code 10H: Set Palette Registers — Causes the interrupt to set the palette registers. This function may be used to set individual palette registers, set the overscan register, set all palette registers and overscan, or set the intensify/blinking bit as follows:

- To set individual palette registers, set register AL to 00H, register BH to the desired palette register, and register BL to the required value.
- To set the overscan register, set register AL to 01H and register BH to the desired value.
- To set all palette registers and overscan, address ES:DX contains a 17-byte table where bytes 0-15 are the palette values and byte 16 is the overscan value.
- To set the intensify/blinking bit, set register AL to 03H and register BL to 00H to enable intensify or 01H to enable blinking.

Function Code 11H: Character Generator Routine — Causes the interrupt to reset the video environment and enter a character generator routine. The functions of this routine are determined by the value you set in register AL as follows:

NOTE: The following functions will not clear the video buffer.

- Set register AL to 00H to load user-specified files. ES:BP is the address of the user table. Set register CX to the number of patterns to store, DX for the character offset into the table, BL for the block to load, and BH to the number of bytes for each character pattern.
- Set register AL to 01H to load ROM monochrome patterns (8×14). Set register BL for the block to load.
- Set register AL to 02H to load ROM double-dot patterns (8×8). Set register BL for the block to load.

Video Interrupts

- Set register AL to 03H to set the block specifier. To set register BL for the character generator block specifier, set bits 3 and 2 to the block for attribute bit 3 = 1 and bits 1 and 0 to the block for attribute bit 3 = 0. Call INT 10H with AX = 1000H and BX = 0712H to set the color plane enable register to ignore bit 3 in addressing color palette registers.

The following functions (AL = 1xH) should be used only after a mode set. They are similar to AL = 0xH with the following exceptions:

- Page 0 must be active.
 - Bytes per character is recalculated.
 - Maximum character rows is recalculated.
 - CRT buffer length is recalculated.
 - CRTC registers are reprogrammed.
- Set register AL to 10H to load user-specified files. ES:BP is the address of the user table. Set register CX to the number of patterns to store, DX for the character offset into the table, BL for the block to load, and BH to the number of bytes for each character pattern.
 - Set register AL to 11H to load ROM monochrome patterns (8 × 14). Set register BL for the block to load.
 - Set register AL to 12H to load ROM double-dot patterns (8 × 8). Set register BL for the block to load.

The following functions (AL = 2xH) should be called only immediately after a mode set.

- Set register AL to 20H for user 8 × 8 graphics characters (INT 1FH). ES:BP is the address of the user table.
- Set register AL to 21H for user graphics characters. ES:BP is the address of the user table. Set register CX for the bytes per character and register BL to the desired number of rows. Refer to Table 11-5 for row specifier value.
- Set register AL to 22H for the ROM 8 × 14 set. Set register BL to the desired number of rows. Refer to Table 11-5 for row specifier values.

Video Interrupts

- Set register AL to 23H for ROM 8 × 8 double dot. Set register BL to the desired number of rows. Refer to Table 11-5 for row specifier values.

Table 11-5. Row Specifier Options

REGISTER BL	NUMBER OF ROWS
00H	User set (DL = number of rows)
01H	14
02H	25
03H	43

- Set register AL to 30H to return information. Register BH must be set to select the information you want returned. Refer to Table 11-6 for available register values and what information they will select.

Table 11-6. Register Values to Return Information for Function 11H

REGISTER BH	INFORMATION RETURNED
00H	INT 1F pointer
01H	INT 44 pointer
02H	ROM 8 × 14 character pointer
03H	ROM 8 × 8 double dot pointer
04H	ROM 8 × 8 double dot (top half) pointer
05H	ROM alternate (9 × 14) pointer

NOTE: When exiting the return information routine (AL = 30H), ES:BP is the specified pointer value, CX is the bytes per character, and DL is the character rows on screen.

Function Code 12H: Alternate Select — Causes the interrupt to return current EGA information or select a routine to set an alternate print screen. Setting register BL to 10H returns current EGA information. Refer to Table 11-7 for interpretation of EGA data. Setting register BL to 20H selects a routine to set up an alternate print screen.

Video Interrupts

Table 11-7. EGA Information Returned by Function Code 12H

REGISTER	VALUE	EXPLANATION
BH	00H	Currently in color mode.
	01H	Currently in monochrome mode (MDA).
BL	00H	64K video memory
	01H	128K video memory
	02H	192K video memory
	03H	256K video memory
CH		Feature bits
CL		Switch setting

Function Code 13H: Write String — Causes the interrupt to write a specified string where ES:BP contains the address of the string to be written. Set register CX for the character count, register DX for the position to begin the string (in cursor terms), and register BH for the page number. Refer to Table 11-8 for further register, string, and cursor information.

Table 11-8. Register, String, and Cursor Data for Function Code 13H

REGISTER AL	REGISTER BL	STRING	CURSOR ACTION
00H	Attribute	"Char ... Char"	No movement
01H	Attribute	"Char ... Char"	Moves
02H	Not used	"Char,Attr...Char,Attr"	No movement
03H	Not used	"Char,Attr...Char,Attr"	Moves

NOTE: Carriage return, line feed, back space, and bell are interpreted as commands, not as printable characters.

Video Interrupts

Function Code 64H: Set Scrolling Mode — Causes the interrupt to select one of three scrolling modes. Place the scrolling mode value in register AL: mode 0 is software scrolling, mode 1 is hardware jump scrolling, and mode 2 is hardware smooth scrolling. Keep in mind the following limitations:

- Hardware scrolling will not work in the 40 × 25 text modes (0 and 1).
- Hardware smooth scrolling works only in the high-resolution graphics mode (6).
- Hardware jump scrolling works only in the graphics modes (4, 5, and 6) and 80 × 25 text mode (3).
- Software scrolling will work in all modes. If you write software that bypasses the Monitor program, use software scrolling.

Video Initialization (INT 1DH)

INT 1DH, unless otherwise programmed by an application program, initializes the video section parameters according to the information stored in the Monitor program ROM. This is the same data used to initialize the video section of the computer when you turn the system on. Table 11-9 defines the values for each register in the 6845 register set. The table shows the relation between each register and the various display modes.

Table 11-9. Video Initialization Default Values

REGISTER NUMBER	TEXT 40 × 25	TEXT 80 × 25	GRAPHICS	MONOCHROME TEXT (TTL)
R0	38	71	38	61
R1	28	50	28	50
R2	2D	5A	2D	52
R3	0A	0A	0A	0F
R4	1F	1F	7F	19
R5	06	06	06	06
R6	19	19	64	19
R7	1C	1C	70	19
R8	02	02	02	02

Table 11-9 (continued). Video Initialization Default Values

REGISTER NUMBER	TEXT 40 × 25	TEXT 80 × 25	GRAPHICS	MONOCHROME TEXT (TTL)
R9	07	07	01	0D
R10	06	06	06	0B
R11	07	07	07	0C
R12	00	00	00	xx
R13	00	00	00	xx
R14	xx	xx	xx	xx
R15	xx	xx	xx	xx
R16	xx	xx	xx	xx
R17	xx	xx	xx	xx

NOTE: All values are expressed in hexadecimal; "xx" represents any value between 00H and FFH.

Defining Characters (INT 1FH)

INT 1FH allows access to an extended character set for use with the medium- and high-resolution color graphics modes. Normally, the character generator ROM supplies the first 128 characters (00H – 7FH) used in the graphics modes. In addition to these characters, you can create a custom character set of 128 additional characters (80H – FFH) using the following procedure:

1. Allocate a 1K section of memory (not video memory) to hold the character set. You will need eight bytes for each character you create.
2. Define each character in an 8 × 8 matrix as shown by the 7 in Figure 11-1. Since the top line in the matrix butts up against the bottom line of the character above it, you will need to allow for ascenders and descenders. Do not forget to allow space between characters.
3. For each line of the matrix, identify which of the eight pixels will be on.
4. With the first pixel as the most-significant bit, add the binary weights of the lit pixels in each row to produce a decimal value for the byte representing that row.

Video Interrupts

5. Load all eight bytes that form the defined character into the first eight bytes of the memory allocated for the character set.
6. Repeat this procedure for each of the 128 characters in the set.
7. Once the characters are defined and placed in memory, set the pointer at interrupt 1FH (memory location 0000:007C) to the start of memory allocated for the defined character set.
8. Execute the INT 1FH instruction. Then, whenever a character code between 80H and FFH is required for printing, the character from your set will be used.

ROW	BINARY VALUE								RESULTING VALUE
	80H	40H	20H	10H	08H	04H	02H	01H	
1									3EH
2									02H
3									02H
4									04H
5									08H
6									10H
7									10H
8									00H

Figure 11-1. Character Design Matrix

EGA Video Considerations

The EGA mode supports a wide range of video modes, including a number of modes also supported by the Z-100 PC standard CGA color video mode. Video modes 0, 1, 2, and 3 offer expanded capabilities in the EGA mode over the standard CGA system. The discussion in this section is divided into three areas: monochrome video modes, normal color video modes, and enhanced color video modes. Each of these apply to operation of the video system in the EGA mode.

Monochrome Video Modes

Table 11-10 describes the attributes of the two possible modes that may be used to generate a monochrome display.

Table 11-10. Monochrome Video Modes

VIDEO MODE NUMBER	DESCRIPTION
7	A monochrome alphanumeric text mode of 80, 9 × 14-pixel characters per line on 25 lines in a screen that has an overall resolution of 720 × 350 pixels. Each character may be displayed in one of four attribute modes. The memory is organized into eight pages and has a base address at B0000H.
F	A monochrome, bit-mapped mode that generates 80, 8 × 14-pixel characters per line on 25 lines in a screen that has an overall resolution of 640 × 350 pixels. Each character may be displayed in one of four attribute modes. The memory is organized into two pages and has a base address at A0000H. This video mode is unique to the EGA card and will not create a display if a Z-100 PC color video card is installed and actively controlling the video output.

Normal Color Video Modes

Table 11-11 describes the attributes of the nine possible normal color modes supported by the EGA card.

Table 11-11. Normal Color Video Modes

VIDEO MODE NUMBER	DESCRIPTION
0	A 16-color alphanumeric mode that generates 40, 8 × 8-pixel characters per line on 25 lines in a screen that has an overall resolution of 320 × 200 pixels.
1	A 16-color alphanumeric mode that generates 40, 8 × 8-pixel characters on 25 lines in a screen that has an overall resolution of 320 × 200 pixels.

Video Interrupts

Table 11-11 (continued). Normal Color Video Modes

VIDEO MODE NUMBER	DESCRIPTION
2	A 16-color alphanumeric mode that generates 80, 8 × 8-pixel characters per line on 25 lines in a screen that has an overall resolution of 640 × 200 pixels.
3	A 16-color alphanumeric mode that generates 80, 8 × 8-pixel characters per line on 25 lines in a screen that has an overall resolution of 640 × 200 pixels.
4	A 4-color, bit-mapped mode that generates 40, 8 × 8-pixel characters per line on 25 lines in a screen that has an overall resolution of 320 × 200 pixels.
5	A 4-color, bit-mapped mode that generates 40, 8 × 8-pixel characters per line on 25 lines in a screen that has an overall resolution of 320 × 200 pixels.
6	A 2-color, bit-mapped mode that generates 80, 8 × 8-pixel characters per line on 25 lines in a screen that has an overall resolution of 640 × 200 pixels.
D	A 16-color, bit-mapped mode that generates 40, 8 × 8-pixel characters per line on 25 lines in a screen that has an overall resolution of 320 × 200 pixels. The memory is organized into eight pages with a base address of A0000H. This mode is unique to the EGA card and will not create a display if a Z-100 PC color video card is installed and actively controlling the video output.
E	A 16-color, bit-mapped mode that generates 80, 8 × 8-pixel characters per line on 25 lines in a screen that has an overall resolution of 640 × 200 pixels. The memory is organized into four pages with a base address of A0000H. This mode is unique to the EGA card and will not create a display if a Z-100 PC color video card is installed and actively controlling the video output.

NOTE: In video modes 0, 1, 2, and 3 the memory is organized into eight pages with a base address of B8000H. In video modes 4, 5, and 6 the memory is organized into one page with a base address of B8000H. Unless specified, all modes are compatible with the display produced by a Z-100 PC color video card.

Enhanced Color Video Modes

Table 11-12 describes the attributes of the five possible enhanced color video modes supported by the EGA card.

Table 11-12. Enhanced Color Video Modes

VIDEO MODE NUMBER	DESCRIPTION
0	A 16-color, alphanumeric mode that generates 40, 8 × 14-pixel characters per line on 25 lines in a screen that has an overall resolution of 320 × 350 pixels.
1	A 16-color, alphanumeric mode that generates 40, 8 × 14-pixel characters per line on 25 lines in a screen that has an overall resolution of 320 × 350 pixels.
2	A 16-color, alphanumeric mode that generates 80, 8 × 14-pixel characters per line on 25 lines in a screen that has an overall resolution of 640 × 350 pixels.
3	A 16-color, alphanumeric mode that generates 80, 8 × 14-pixel characters per line on 25 lines in a screen that has an overall resolution of 640 × 350 pixels.
10	A 16-color, bit-mapped mode that generates 80, 8 × 14-pixel characters per line on 25 lines in a screen that has an overall resolution of 640 × 350 pixels. The memory is organized into two pages with a base address of A0000H. The 16 colors can be selected from a palette of 64 colors.

NOTE: In video modes 0, 1, 2, and 3 the memory is organized into eight pages with a base address of B8000H. The 16 colors can be selected from a palette of 64 colors.

Basic Modes of Operation

The 256K of video memory is divided into four separately addressable, but parallel, 64K memory planes. Depending upon the video mode selected, this memory may be combined in different ways. The following discussions describe the basic modes of operation and how they work with the video memory.

Text (Alphanumeric) Modes

In the text modes a two-byte character/attribute format is used to define each character display position on the screen. These two bytes are mapped into assigned locations in video memory with the character byte in memory plane 0 and the attribute byte in memory plane 1.

Normally, bit 3 of the attribute byte is used by the video cards to determine the intensity characteristic of the character being displayed. However, the expanded video card allows bit 3 to be re-defined by the character map select register and used as a switch between character sets. This gives the programmer access to both character sets, for a total 512 characters, instead of just one character set. To accommodate the additional 256 characters, the patterns of both character generators are transferred into memory plane 2 of the video memory. In addition, with 256K of video memory on this card, it is possible to support two additional user-defined, 256-member character sets for a total of 1024 different characters. The patterns of these additional character sets must also be loaded into memory plane 2 of the video memory.

To display a screen of data, the CRT controller generates a series of sequential addresses, which represents the sequential locations of characters on the screen. These addresses correspond to video memory addresses where memory planes 0 and 1 are accessed. The byte in memory plane 0, combined with the scan row count, determines the address of the bit pattern stored in memory plane 2. If bit 3 of the attribute byte has been selected to act as a switch between character sets, the video memory address is modified accordingly. The bit pattern found in memory plane 2 is then assigned the attributes found in the attribute byte and sent as a serial stream of data out the appropriate pins of the video connector.

Monochrome and color attribute data is formatted differently. Table 11-13 describes the attribute byte characteristics for monochrome modes.

Table 11-13. Monochrome Attribute Byte Characteristics

BIT									ATTRIBUTE DESCRIPTION
	7	6	5	4	3	2	1	0	
B	0	0	0	1	1	1	1	1	Normal video
B	1	1	1	1	0	0	0	0	Reverse video
B	0	0	0	1	0	0	0	0	No display (black output)
B	1	1	1	1	1	1	1	1	No display (white output)

NOTE: Bit 7 defines whether the character blinks on and off. If bit 7 is a 1, the character will blink. If bit 7 is a 0, the character will not blink. Bit 3 defines either the intensity attribute or the character set. Character set definition is determined by the character map select register. If bit 3 controls the intensity attribute, then a 1 will cause the character to be displayed at high intensity and a 0 will cause the character to be displayed at normal intensity.

Table 11-14 describes the attribute byte characteristics for color modes. The bit pattern for the foreground color and background color are identical. The foreground color attribute occupies bits 0 – 2 and background color attribute occupies bits 4 – 6.

Table 11-14. Color Attribute Byte Characteristics

BACKGROUND			FOREGROUND			NORMAL COLOR	INTENSIFIED COLOR
BIT			BIT				
6	5	4	2	1	0		
0	0	0	0	0	0	Black	Dark gray
0	0	1	0	0	1	Blue	Light blue
0	1	0	0	1	0	Green	Light green
0	1	1	0	1	1	Cyan	Light cyan
1	0	0	1	0	0	Red	Light red (pink)
1	0	1	1	0	1	Magenta	Light magenta
1	1	0	1	1	0	Brown	Yellow
1	1	1	1	1	1	White	Intensified white

NOTE: Bit 7 defines whether the character blinks on and off. If bit 7 is a 1, the character will blink. If bit 7 is a 0, the character will not blink. Bit 3 defines either the intensity attribute or the character set. Character set definition is determined by the character map select register. If bit 3 controls the intensity attribute, then a 1 will cause the character to be displayed at high intensity and a 0 will cause the character to be displayed at normal intensity.

Video Interrupts

Graphics Modes

With the exception of video mode F (high-resolution monochrome) and video mode 10 (enhanced color graphics), addressing, mapping, and data formatting are the same as those used with Z-100 PC color video cards and PC-equivalent color/graphics adapters.

In the Z-100 PC color video graphics modes, two degrees of resolution are available: 320×200 and 640×200 . 640×200 is obtainable only as a monochrome, or two-color, display (one of 16 border colors may be used for the displayed pixels). In the 320×200 modes, each displayed pixel may be one of four colors selected from two palettes and the border color.

Medium-Resolution Color

Pixel information for the 320×200 display is stored in two memory planes of 8000 bytes each, starting at address B8000H. The even-numbered (0, 2, 4, ... 198) scan row information is stored from B8000H to B9F3FH. The odd-numbered (1, 3, 5, ... 199) scan row information is stored from BA000H to BBF3FH.

Each byte of video memory contains four two-bit pairs, which define the color displayed for that pixel. Note that there is no unlit pixel capability; if you want to unlight any given pixel, you will have to sacrifice one of the possible colors (the background color) to black.

Table 11-15 defines the color selected for the defined pixel. Since only one palette can be active, Table 11-16 defines the colors in the two possible palettes for these video modes.

Table 11-15. Color Selection

MSB	LSB	COLOR DEFINITION
0	0	The pixel will display the current background color.
0	1	The pixel will display color 1 of the current palette.
1	0	The pixel will display color 2 of the current palette.
1	1	The pixel will display color 3 of the current palette.

Table 11-16. Palette Colors

COLOR NUMBER	PALETTE 1	PALETTE 2
1	Cyan	Green
2	Magenta	Red
3	White	Brown

The 16 available border colors are the same that can be used to display text characters: black, blue, green, cyan, red, magenta, brown, white, dark gray, light blue, light green, light cyan, light red (pink), light magenta, yellow, and intensified white.

High-Resolution Color

The high-resolution color mode is actually considered a monochrome display mode. The entire memory complement in a Z-100 PC color video card (16K) is needed to define the on or off states of a full screen of pixels. Each pixel location on the 640×200 screen is represented by a location in video memory, starting at B8000H. Because each location can be set to only a 1 or a 0, each pixel may be the border color or not lit. The 16 border colors are: black, blue, green, cyan, red, magenta, brown, white, dark gray, light blue, light green, light cyan, light red (pink), light magenta, yellow, and intensified white.

Mode F

Mode F is a high-resolution 640×350 mode that supports monochrome black and white pixel graphics with the following attributes: black, video, blinking video, and intensified video. Two memory planes, required to support the four attributes, are both addressed at A0000H by the CPU. The first memory plane is identified as the video memory plane and the second is identified as the intensity memory plane. Actually, a combination of pixel pairs between the two memory planes is required to define the attributes, as shown in Table 11-17.

Video Interrupts

Table 11-17. Mode F Attributes

ATTRIBUTE	VIDEO MEMORY PLANE	INTENSITY MEMORY PLANE
Black (not lit)	0	0
Video (lit)	0	1
Blinking video	1	0
Intensified video	1	1

NOTE: Some IBM-equivalent enhanced graphics adapters are supplied with only 64K of video memory. When this is the case, memory planes 0 and 1 and memory planes 2 and 3 are mapped together to form the two 32K memory planes needed to support this mode. Although the four memory planes all reside at an A0000H starting address, memory planes 0 and 2 are addressed as even memory planes and memory planes 1 and 3 are addressed as odd memory planes. When data is brought out of video memory for display, the first byte comes out of the even memory planes, the second out of the odd memory planes, the third out of the even, and so on.

In this EGA card there is 256K of memory, so the memory planes are not mapped together as described in the preceding note. The first (memory plane 0) and third (memory plane 2) are used for mode F. The second (memory plane 1) and fourth (memory plane 3) are ignored.

Two bits, one from each plane, define each pixel on the screen. Since memory organization is sequential, the first eight pixels are contained in the first byte, starting at video memory address A0000H and with the most significant bit. The second byte is located at video memory address A0001H and so on.

Mode 10

Mode 10 supports graphics in 16 colors, selected from 64 possibilities. All four available memory planes in video memory are used. Each memory plane normally represents a color or intensity, as shown in Table 11-18. The base colors that can be produced without programming additional registers are described in Table 11-19. The other possible shades are obtained by programming the palette registers in the attribute controller.

Table 11-18. Mode 10 Memory Plane Assignments

MEMORY PLANE	COLOR OR FUNCTION
0	Blue
1	Green
2	Red
3	Intensified

Table 11-19. Mode 10 Base Colors

BP3	BP2	BP1	BP0	RESULTING COLOR
0	0	0	0	Black
0	0	0	1	Blue
0	0	1	0	Green
0	0	1	1	Cyan
0	1	0	0	Red
0	1	0	1	Magenta
0	1	1	0	Brown
0	1	1	1	White
1	0	0	0	Dark gray
1	0	0	1	Light blue
1	0	1	0	Light green
1	0	1	1	Light cyan
1	1	0	0	Light red
1	1	0	1	Light magenta
1	1	1	0	Yellow
1	1	1	1	Intensified white

Video Interrupts

There are 16 palette registers that correspond to the 16 possible base colors. The registers allow you to remap the color represented by the bit patterns shown in the preceding table to any one of 64 possible colors that can result from combining the six bits in each register. Refer to Table 11-20 for the bit-to-color relationship. Setting a bit to 1 activates that color. Setting the bit to 0 deactivates that color.

Table 11-20. Palette Register Color Attributes

BIT	COLOR
0	Blue
1	Green
2	Red
3	Secondary blue
4	Secondary green
5	Secondary red

Register Default Values

The various registers in the EGA design have different values, depending on the video mode. Table 11-21 defines the external register values. Tables 11-22 through 11-25 define the register values of the 82C431, 82C432, 82C433, and 82C434, respectively. These LSI devices make up a specialized chip set that performs the major video functions on the 31 kHz card. Refer to Chapter 17 for a detailed discussion of these devices.

NOTE: Tables 11-21 through 11-25 define the default register values for the video chip set when it is configured for standard EGA operation. These values are not the same default values as those used for 31 kHz operation.

Table 11-21. External Register Values

REGISTER NAME	PORT ADDRESS	MODE OF OPERATION											
		0	1	2	3	4	5	6	7	D	E	F	10
Misc	3C2	A7	A7	A7	A7	23	23	23	A6	23	23	A2	A7
Feature	3?A	00	00	00	00	00	00	00	00	00	00	00	00
Inp St 1	3C2	—	—	—	—	—	—	—	—	—	—	—	—
Inp St 2	3?2	—	—	—	—	—	—	—	—	—	—	—	—

NOTES: ? = B in monochrome modes, D in color modes
 — = "don't care"

Table 11-22. 82C431 Register Values

REGISTER NAME	PORT ADDRESS	MODE OF OPERATION											
		0	1	2	3	4	5	6	7	D	E	F	10
Grphcs A	3CC	00	00	00	00	00	00	00	00	00	00	00	00
Grphcs B	3CA	01	01	01	01	01	01	01	01	01	01	01	01
Address	3CE	—	—	—	—	—	—	—	—	—	—	—	—
Set-reset	3CF-00	00	00	00	00	00	00	00	00	00	00	00	00
En St/rst	3CF-01	00	00	00	00	00	00	00	00	00	00	00	00
Color cmp	3CF-02	00	00	00	00	00	00	00	00	00	00	00	00
Data rot	3CF-03	00	00	00	00	00	00	00	00	00	00	00	00
Rd map sl	3CF-04	00	00	00	00	00	00	00	00	00	00	00	00
Mode reg	3CF-05	10	10	10	10	30	30	00	10	00	00	00	00
Misc	3CF-06	0E	0E	0E	0E	0F	0F	0D	0A	05	05	05	05
Clr dt cr	3CF-07	00	00	00	00	00	00	00	00	00	00	00	00
Bit mask	3CF-08	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF

NOTES: ? = B in monochrome modes, D in color modes
 — = "don't care"

Video Interrupts

Table 11-23. 82C432 Register Values

REGISTER NAME	PORT ADDRESS	MODE OF OPERATION										
		0	1	2	3	4	5	6	7	D	E	F 10
Address	3C4	—	—	—	—	—	—	—	—	—	—	—
Reset	3C5-00	0B	0B	03	03	03	03	03	03	03	03	03
Clock Md	3C5-01	0B	0B	01	01	0B	0B	01	00	0B	01	01
Map Mask	3C5-02	03	03	03	03	03	03	01	03	0F	0F	0F
Char Gen	3C5-03	00	00	00	00	00	00	00	00	00	00	00
Mem Mode	3C5-04	03	03	03	03	02	02	06	03	06	06	06

NOTES: ? = B in monochrome modes, D in color modes

— = "don't care"

Table 11-24. 82C433 Register Values

REGISTER NAME	PORT ADDRESS	MODE OF OPERATION										
		0	1	2	3	4	5	6	7	D	E	F 10
Address	3C0	—	—	—	—	—	—	—	—	—	—	—
Palette	3C0-00	00	00	00	00	00	00	00	00	00	00	00
Palette	3C0-01	01	01	01	01	13	13	17	08	01	01	08
Palette	3C0-02	02	02	02	02	15	15	17	08	02	02	00
Palette	3C0-03	03	03	03	03	17	17	17	08	03	03	00
Palette	3C0-04	04	04	04	04	02	02	17	08	04	04	18
Palette	3C0-05	05	05	05	05	04	04	17	08	05	05	18
Palette	3C0-06	14	14	14	14	06	06	17	08	06	06	00
Palette	3C0-07	07	07	07	07	07	07	17	08	07	07	00
Palette	3C0-08	38	38	38	38	10	10	17	10	10	10	00
Palette	3C0-09	39	39	39	39	11	11	17	18	11	11	08
Palette	3C0-0A	3A	3A	3A	3A	12	12	17	18	12	12	00
Palette	3C0-0B	3B	3B	3B	3B	13	13	17	18	13	13	00
Palette	3C0-0C	3C	3C	3C	3C	14	14	17	18	14	14	00
Palette	3C0-0D	3D	3D	3D	3D	15	15	17	18	15	15	18
Palette	3C0-0E	3E	3E	3E	3E	16	16	17	18	16	16	00
Palette	3C0-0F	3F	3F	3F	3F	17	17	18	17	17	00	3F
Mode ctrl	3C0-10	08	08	08	08	01	01	01	0E	01	01	0B
Overscan	3C0-11	00	00	00	00	00	00	00	00	00	00	00
Clr plane	3C0-12	0F	0F	0F	0F	03	03	01	0F	0F	0F	05
Hrz pan	3C0-13	00	00	00	00	00	00	00	00	00	00	00

NOTES: ? = B in monochrome modes, D in color modes

— = "don't care"

Video Interrupts

Table 11-25. 82C434 Register Values

REGISTER NAME	PORT ADDRESS	MODE OF OPERATION											
		0	1	2	3	4	5	6	7	D	E	F	10
Address	3?4	—	—	—	—	—	—	—	—	—	—	—	—
Hz total	3?5-00	2D	2D	5B	5B	37	37	70	60	37	70	60	5B
Hz dsply	3?5-01	27	27	4F	4F	27	27	4F	4F	27	4F	4F	4F
S hrz blk	3?5-02	2B	2B	53	53	2D	2D	59	56	2D	56	56	53
E hrz blk	3?5-03	2D	2D	37	37	37	37	2D	3A	37	2D	3A	37
S hrz rtc	3?5-04	28	28	51	51	30	30	5E	51	30	5E	50	5E
E hrz rtc	3?5-05	6D	6D	5B	5B	14	14	06	60	14	06	60	00
Vert totl	3?5-06	6C	6C	6C	6C	04	04	04	70	04	04	70	6C
Overflow	3?5-07	1F	1F	1F	1F	11	11	11	1F	11	11	1F	1F
Prst row	3?5-08	00	00	00	00	00	00	00	00	00	00	00	00
Mx scn ln	3?5-09	0D	0D	0D	0D	01	01	01	0D	00	00	00	00
Cursor s	3?5-0A	0B	0B	0B	0B	00	00	00	0B	00	00	00	00
Cursor e	3?5-0B	0C	0C	0C	0C	00	00	00	0C	00	00	00	00
S addr h	3?5-0C	—	—	—	—	—	—	—	—	—	—	—	—
S addr l3	3?5-0D2	—	—	—	—	—	—	—	—	—	—	—	—
Cursor h3?	5-0E2	—	—	—	—	—	—	—	—	—	—	—	—
Cursor l3?	5-0F2	—	—	—	—	—	—	—	—	—	—	—	—
V rtc s	3?5-10	5E	5E	5E	5E	E1	E1	E0	5E	E1	E0	5E	5E
V rtc e	3?5-11	2B	2B	2B	2B	24	24	23	2E	24	23	2E	2B
L pen h	3?5-10	—	—	—	—	—	—	—	—	—	—	—	—
L pen l	3?5-11	—	—	—	—	—	—	—	—	—	—	—	—
V dsply e	3?5-12	5D	5D	5D	5D	C7	C7	C7	5D	C7	C7	5D	5D
Offset	3?5-13	14	14	28	28	14	14	28	28	14	28	28	28
Uline loc	3?5-14	0F	0F	0F	0F	00	00	00	0D	00	00	0D	0F
S v blk	3?5-15	5E	5E	5E	5E	E0	E0	DF	5E	E0	DF	5E	5F
E v blk	3?5-16	0A	0A	0A	0A	F0	F0	EF	6E	F0	EF	6E	0A
Md ctrl	3?5-17	A3	A3	A3	A3	A2	A2	C2	A3	E3	E3	E3	E3
Ln cmp	3?5-18	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF

NOTES: ? = B in monochrome modes, D in color modes

— = "don't care"



Part III

System Hardware



Chapter 12

The CPU

This chapter explains the programmer accessible features of the Intel 80386 microprocessor. This device is one of the most advanced microprocessor products in use today.

Introduction

The system CPU card, located in slot 4 of the backplane card, contains the majority of the computer system's control circuitry. The card contains the 80386 CPU, sockets for either the 80287 or the 80387 coprocessors, the system firmware, and a socket for an additional ROM device. The card also contains address and data bus buffers, system decoding and control logic, 32-bit memory control logic, the system clock circuits and the interface for the optional cache controller. Figure 12-1 is a block diagram of the CPU card showing the organization of this card. Only the microprocessors located on the CPU card are programmable by the user.

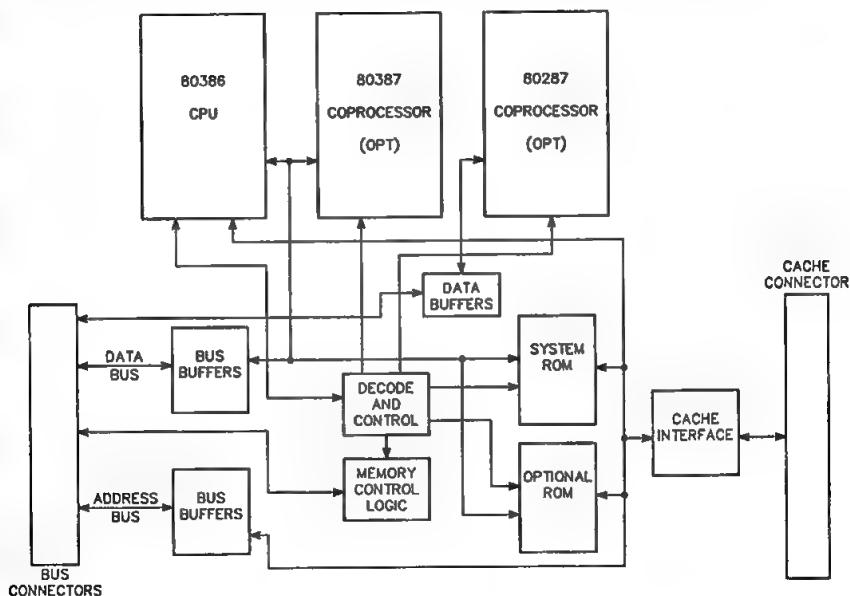


Figure 12-1. CPU Card Block Diagram

The CPU

The 80386 is a full 32-bit microprocessor employing a 32-bit data path internally as well as externally. The design of the 80386 is an extension of the 8086 and the 80286 microprocessors previously developed by Intel. The 80386 instruction set (refer to Appendix A) is a superset of the instruction sets of these earlier microprocessors. This provides a great deal of flexibility by permitting programs developed for the earlier devices to run on the 80386. The 80386 differs from the earlier 8086 and the 80286 only in the extent of its capabilities. The processor maintains the instruction sets and capabilities of these earlier devices. It extends those capabilities by adding more instructions, features, and raw speed to the computing environment.

80386 Base Architecture

This section describes the base architecture of the 80386. For the purposes of this discussion the base architecture refers to the general internal architecture of the processor.

NOTE: The processor architecture alters somewhat when the device switches between protected mode or virtual 8086 mode. Later sections of this chapter contain discussions of these modes.

Internally, the 80386 consists of three main units, the central processing unit, the memory management unit, and the bus interface. The central processing unit consists of the execution unit and the instruction unit, which are very similar to the earlier 8086 and 80286 microprocessors. However, the 80386 also includes a memory management unit with virtual page capability. The memory management unit consists of a segmentation unit and a paging unit. Figure 12-2 is a block diagram of the 80386 architecture.

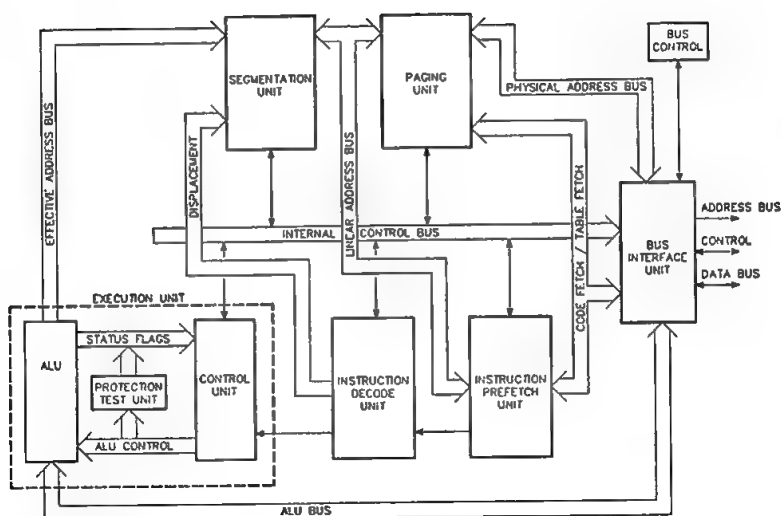


Figure 12-2. 80386 Processor Architecture

Bus Interface Unit — The bus interface unit (BIU) provides the interface between the 80386 microprocessor and the rest of the computer. This unit accepts requests for code fetches and data transfers and provides priorities for these requests. This unit also provides the control signals required for execution of the current bus cycle and controls the interface to the system coprocessor.

Instruction Prefetch Unit — The prefetch unit essentially performs a program look ahead function for the 80386. This unit monitors the bus interface unit. When the BIU is not performing data transfers, this unit uses the BIU to sequentially prefetch instructions. The BIU has a 16-byte queue it uses to store these instructions. Since prefetch operations have a lower processor priority than data transfers, the activity of the prefetch unit does not degrade processor performance.

Instruction Decode Unit — This unit accepts the instructions from the prefetch unit and translates them into microcode for the processor. This unit has a FIFO buffer capable of storing three decoded instructions. This unit also generates immediate data and opcode offsets for logical addresses.

The CPU

Execution Unit — The execution unit consists of the ALU, the control unit, the protection test unit, and eight 32-bit general purpose registers. This unit accepts the decoded instructions and executes them. In doing this it also communicates with other portions of the microprocessor to obtain required data.

Segmentation Unit — The segmentation unit converts the logical addresses into linear addresses under the control of the execution unit. The segmentation unit also checks for segmentation violations. The processor uses an on-chip cache to help speed up these translations.

Paging Unit — If the paging mode is active in the 80386, this unit will translate linear addresses into physical addresses. If the paging mode is not active, no translation takes place. The paging unit also uses an on-chip cache to speed up translations. The processor sends the physical addresses to the BIU for memory and I/O accesses.

Register Resources

The 80386 microprocessor contains 32 programmer-accessible registers. These registers can be grouped according to their functions, as follows:

- General purpose registers
- Instruction pointer and flags
- Segment registers
- Control registers
- Address registers
- Debug registers
- Test registers.

The CPU

Execution Unit — The execution unit consists of the ALU, the control unit, the protection test unit, and eight 32-bit general purpose registers. This unit accepts the decoded instructions and executes them. In doing this it also communicates with other portions of the microprocessor to obtain required data.

Segmentation Unit — The segmentation unit converts the logical addresses into linear addresses under the control of the execution unit. The segmentation unit also checks for segmentation violations. The processor uses an on-chip cache to help speed up these translations.

Paging Unit — If the paging mode is active in the 80386, this unit will translate linear addresses into physical addresses. If the paging mode is not active, no translation takes place. The paging unit also uses an on-chip cache to speed up translations. The processor sends the physical addresses to the BIU for memory and I/O accesses.

Register Resources

The 80386 microprocessor contains 32 programmer-accessible registers. These registers can be grouped according to their functions, as follows:

- General purpose registers
- Instruction pointer and flags
- Segment registers
- Control registers
- Address register
- Debug register
- Test register

The design of the registers within the 80386 maintains compatibility with previous Intel processors. The registers in the 80386 are a superset of the registers contained in the 8086 and 80286. Table 12-1 provides an overall graphic view of the various registers and their uses in the 80386 while figure 12-3 illustrates the internal register organization.

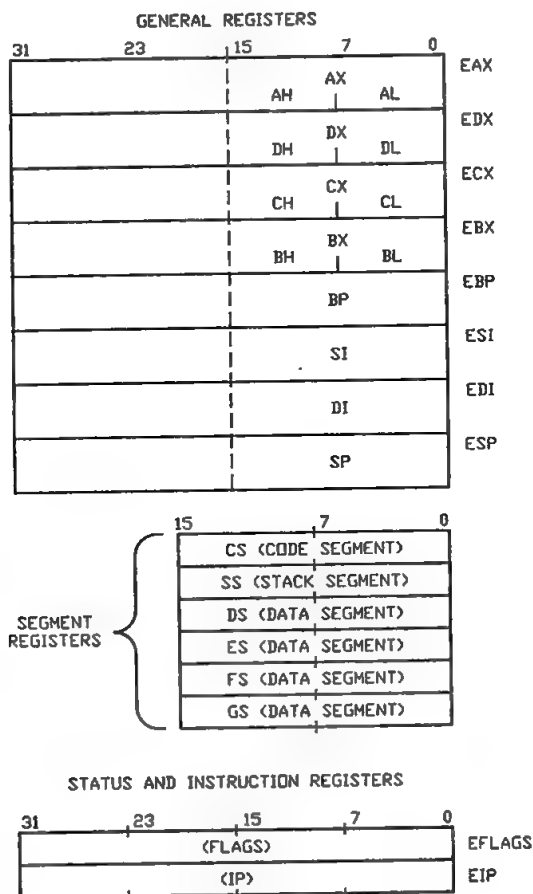


Figure 12-3. 80386 Internal Register Structure

The CPU

Table 12-1. 80386 Register Usage

REGISTER	USAGE MODE					
	REAL		PROTECTED		VIRTUAL	
	LOAD	STORE	LOAD	STORE	LOAD	STORE
General	Y	Y	Y	Y	Y	Y
Segment	Y	Y	Y	Y	Y	Y
Flag	Y	Y	Y	Y	I/O	I/O
Control	Y	Y	P	P	N	Y
GDTR	Y	Y	P	Y	N	Y
IDTR	Y	Y	P	Y	N	Y
LDTR	N	N	P	Y	N	N
TR	N	N	P	Y	N	N
Debug	Y	Y	P	P	N	N
Test	Y	Y	P	P	N	N

NOTES: I/O This indicates that the PUSHF and POPF instructions are I/O privilege level sensitive in this mode.

P This indicates that the registers can be accessed only when the current privilege level is equal to zero.

The registers illustrated in Figure 12-3 are task specific. The processor automatically loads the registers with new contents whenever the processor executes a task switch operation. Table 12-2 describes the functions of these registers.

Table 12-2. 80386 Internal Registers

REGISTER	DESCRIPTION
EAX	32-bit general purpose register containing the AX, AH, and AL registers. Normally operates as the 32-bit accumulator
AX	16-bit accumulator register (least-significant 16 bits of register EAX). This register consists of the 8-bit AH (high) and AL (low) registers. Specifically used by word multiply, word divide, and word input/output instructions.
AH	The high eight bits of register AX. Specifically used by 16-bit byte multiply and byte divide instructions.
AL	The low eight bits of register AX. Specifically used by 16-bit byte multiply, byte divide, byte input/output, translate, and decimal arithmetic instructions.

Table 12-2 (continued). 80386 Internal Registers

REGISTER	DESCRIPTION
EBX	32-bit general purpose register containing the BX, BH, and BL registers. Functions as the 32-bit base register.
BX	16-bit base register (least-significant 16 bits of register EBX). This register consists of the 8-bit BH (high) and BL (low) registers. Specifically used by 16-bit translate instructions.
BH	The high eight bits of register BX.
BL	The low eight bits of register BX.
ECX	32-bit general purpose register containing the CX, CH, and CL registers.
CX	16-bit count register (least-significant 16 bit of register ECX). This register consists of the 8-bit CH (high) and CL (low) registers. Specifically used by 16-bit string operations and loops.
CH	The high eight bits of register CX.
CL	The low eight bits of register CX. Specifically used by variable shift and rotate instructions.
EDX	32-bit general purpose register containing the DX, DH, and DL registers.
DX	16-bit data register (least-significant 16 bits of register EDX). This register consists of the 8-bit DH (high) and DL (low) registers. Specifically used by 16-bit word multiply, word divide, and indirect input/output instructions.
DH	The high eight bits of register DX.
DL	The low eight bits of register DX.
ESI	32-bit general purpose register containing the SI register.
SI	16-bit source index register (least-significant 16 bits of register ESI). The processor uses this 16-bit register for string operations.
EDI	32-bit general purpose register containing the DI register.
DI	16-bit destination index register (least-significant 16 bits of register EDI). The processor also uses this 16-bit register for string operations.
EBP	32-bit general purpose register containing the BP register.
BP	16-bit base pointer register (least-significant 16 bits of register EBP). Although the processor does not use this 16-bit register for any specific operation, it does come into use with certain addressing modes.
ESP	32-bit general purpose register containing the SP register.
SP	16-bit stack pointer register (least-significant 16 bits of register ESP). The processor uses this 16-bit register for stack operations. The register contains the current stack address.

The CPU

Table 12-2 (continued). 80386 Internal Registers

REGISTER	DESCRIPTION
CS	Code segment register. This 16-bit register points to the base address of the current code segment. The CPU fetches instructions from this segment.
SS	Stack segment register. This 16-bit register points to the base address of the current stack segment. This segment holds the stack for the CPU.
DS	Data segment register. This 16-bit register points to the base address of the current data segment. This register normally contains contains variables for the program being executed.
ES	16-bit data segment register.
FS	16-bit data segment register.
GS	16-bit data segment register.
EIP	32-bit register containing the offset, relative to the base of the current code segment, of the next sequential instruction to be executed. This register is not directly visible to the programmer. The low-order 16 bits include the 16-bit IP register.
IP	16-bit instruction pointer (least-significant 16 bits of register EIP). This 16-bit register is similar in purpose to the program counter in 8-bit CPU designs. The contents of this register point to the next instruction location. The processor updates this information and fetches the next instruction through the bus interface unit. Saving the instruction pointer on the stack will insure that the information will remain current.
EFLAGS	32-bit flag register containing the 16-bit register FLAGS
FLAGS	16-bit flag register (least-significant 16 bits of register EFLAGS) for 16-bit operations.

NOTE: All eight 16-bit general purpose registers fit the definition of accumulator as defined for 8-bit CPUs.

General Purpose Registers

The general purpose registers (EAX, EBX, ECX, EDX, ESI, EDI, EBP, and ESP) are 32-bit registers. As shown in Figure 12-3 and Table 12-1 they can also be divided into 16-bit and 8-bit sub-units. The registers can hold either data or address information. They support data operands of 1, 8, 16, 32, or 64 bits and bit fields from 1 to 32 bits long. The registers also support address operands of 16 or 32 bits.

Instruction Pointer and Flags

The instruction pointer (EIP) contains the offset of the next instruction requiring execution. The offset always remains relative to the base of the code segment (CS).

The flags register in the 80386 is a 32-bit register containing the 16-bit FLAGS register. Two new registers exist in the 80386 in addition to those used in the 80286. Figure 12-4 illustrates these additions. Table 12-3 describes the 80386 flag set. You should not use the reserved flag bits in the 80386 for systems or applications programs.



Figure 12-4. 80386 Flags Register

Table 12-3. 80386 Flag Definitions

FLAG BIT	PURPOSE
VM (bit 17)	Virtual 8086 mode. If set while the processor is in protected mode, the 80386 will switch to virtual 8086 operation.
RF (bit 16)	Resume flag. Use this flag in conjunction with the debug registers. If this bit is active, the processor will ignore any debug fault occurring on the next instruction.

The CPU

Table 12-3 (continued). 80386 Flag Definitions

FLAG BIT	PURPOSE
NT (bit 14)	Nested task. This flag deals with protected mode operations. In the 80386 a task may be nested within another task. A set condition for this flag indicates that the current task has a valid back link to the previous task.
IOPL (bit 12-13)	Input/output privilege level. This flag also pertains to protected mode operations. This flag indicates the maximum current privilege level (CPL) permitted for I/O instruction execution.
OF (bit 11)	Overflow flag. This flag is set if an operation results in a signed overflow. A signed overflow condition exists if an operation causes a carry/borrow into the sign bit without a corresponding carry/borrow out of the high-order bit, or vice versa.
DF (bit 10)	Direction flag. If set, the ESI and/or EDI registers will postdecrement during string operations. If not, the registers will postincrement.
IF (bit 9)	INTR enable flag. If set, the processor will recognize external interrupts on the INTR pin.
TF (bit 8)	Trap enable flag. If set, the 80386 will generate an exception 1 trap after the next instruction executes.
SF (bit 7)	Sign flag. This flag sets if the high-order bit of the operation results in a high logic state (1).
ZF (bit 6)	Zero flag. This flag is set if all bits of the result are zero.
AF (bit 4)	Auxiliary carry flag. This flag is set if the addition of a packed BCD value resulted in a carry out of bit 3. It is also set if the subtraction of a packed BCD value resulted in a borrow from bit 3. This flag is only affected by the operation on bit 3, regardless of the overall operand length.
PF (bit 2)	Parity flag. This flag is set if the low-order eight bits of an operation contains an even number of 1s. Only the value of the low-order eight bits affects the parity flag.
CF (bit 0)	Carry flag. This flag is set for addition if the operation resulted in a carry out of the high order bit. A subtraction operation resulting in a borrow out of the high-order bit will also set the flag.

Segment Registers

There are six 16-bit registers within the 80386. These registers are available for holding segment selector values to identify the currently addressable memory segments. The available segment size varies with the mode of operation for the 80386. In protected mode the segment size can vary from one byte up to the maximum memory size of the computer (16M). Real address mode restricts the segment size to a maximum of 64K. The CS register maintains the current code segment information. The SS register maintains the current stack segment information. The remaining four registers are available for storing information related to data storage. There is an additional register associated with each segment register in the 80386. These registers are the segment descriptor registers.

The segment descriptor registers are not totally visible to the programmer; however, their function is closely tied to the programming task. Each descriptor register contains a 32-bit segment base address, a 20-bit segment limit, and segment attributes. When the processor loads a value into a segment register, it automatically updates the descriptor register. In real address mode the base address is the only portion of the descriptor that the processor updates. This is because the segment limits and attributes remain fixed in real mode. In protected mode the processor updates the entire descriptor register.

Whenever a memory reference occurs, the processor automatically involves the segment descriptor register with the reference. The descriptor register's 32-bit segment base address becomes a part of the linear address calculation, the limit serves in limit check operations. The system checks the attributes against the type of reference requested.

The CPU

Control Registers

The 80386 employs four 32-bit control registers to maintain the current global machine state. Three of these registers are accessible to the programmer. Figure 12-5 illustrates the 80386 control registers.

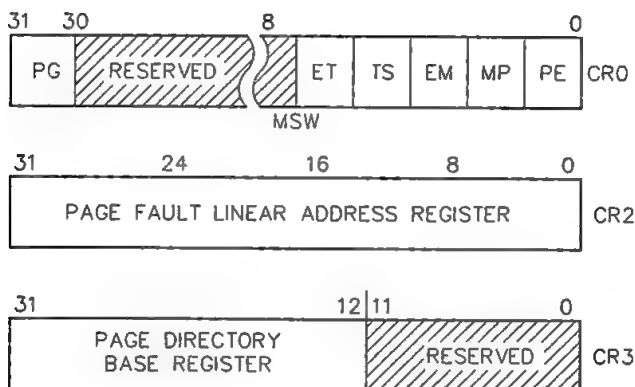


Figure 12-5. 80386 Control Registers

The CR0 register is the machine control register. It contains six bits that define machine status and control modes. In order to maintain compatibility with 80286 protected mode, the low-order 16 bits of CR0 are known as the machine status word. The special LMSW and SMSW instructions (refer to Appendix A) handle these 16 bits the same in the 80386 as they do in the 80286. Intel recommends the use of the MOV CR0, REG class instruction to change bits that the LMSW instruction does not affect. You should not use the reserved bits in this or any other register for programming operations. Table 12-4 details the defined bit functions in CR0.

Table 12-4. Register CR0 Bit Definitions

BIT	FUNCTION
PG (bit 31)	Paging enable. Setting this bit enables the on-chip paging unit.
ET (bit 4)	Processor extension type. If set, this bit indicates that a 80387 coprocessor is present and the processor will use 32-bit protocol. If clear, the processor will expect a 80287 coprocessor and will use 16-bit protocol. For strict 80286 compatibility the LMSW instruction does not affect this bit.
TS (bit 3)	Task switch. Whenever a task switch operation occurs, the processor sets this bit. A coprocessor escape opcode will cause a "coprocessor not available" trap. The trap handler will save the previous task's 80287/80387 context, load the 80287/80387 context for the current task, and clear the TS bit before returning to the faulting opcode.
EM (bit 2)	Emulate coprocessor. If set, this bit will cause all coprocessor opcodes to generate a "coprocessor not available" fault. The setting of this bit does not affect the coprocessor WAIT opcode.
MP (bit 1)	Monitor coprocessor. Use this bit with the TS bit to determine if the WAIT opcode will generate a "coprocessor not available" fault when $TS = 1$. If the TS and MP bits are both set, the opcode will generate a trap.
PE (bit 0)	Protection enable. Setting this bit enables protected mode in the processor. If the bit is clear, the processor will operate in real mode. Loading the machine status word or CR0 will set this bit. The only way to clear this bit is to load CR0. This allows the 80386 to maintain compatibility with the 80286.

The CPU

The CR2 register is the page fault linear address register. This register contains the 32-bit linear address that caused the last detected page fault. You can find additional information about the error in the error code pushed onto the page fault handler's stack.

The CR3 register is the page directory base address register. This register contains the physical base address of the page directory table. In the 80386 this table is always page-aligned on 4K boundaries. This means that the processor ignores the lowest 12 bits of this register. These are reserved bits and should not be used.

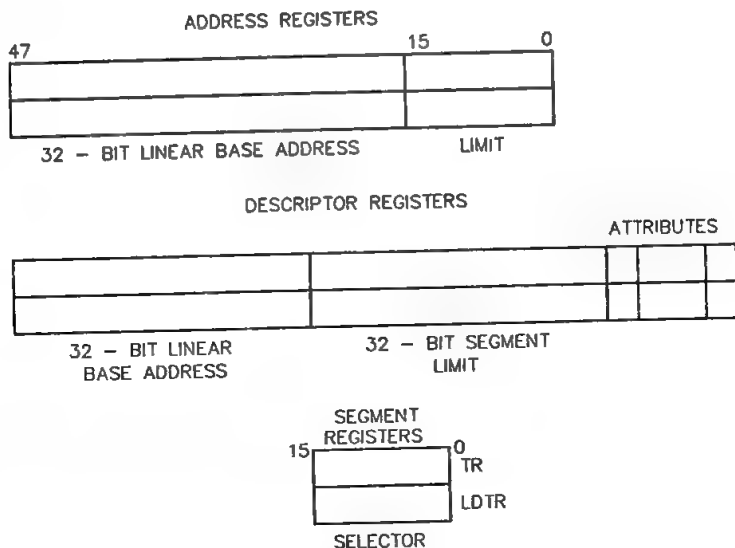
If a task switch operation changes the value of CR2, or CR3 is explicitly loaded with any value, all cached page table entries in the paging unit cache are no longer valid.

Address Registers

The 80386 supports four special registers to refer to tables or segments in protected mode. These registers support similar operations in the earlier 80286 processor. The tables or segments referred to are:

- TSS (task state segment)
- LDT (local descriptor table)
- GDT (global descriptor table)
- IDT (interrupt descriptor table).

The CPU uses special registers to store these values. These are the system address and system segment registers referred to as TR, LDTR, GDTR and IDTR, respectively. Figure 12-6 illustrates these registers.



Reprinted by permission of Intel Corporation, Copyright 1987

Figure 12-6. 80386 System Segment Registers

The GDTR and IDTR registers hold the 32-bit linear base address and the 16-bit limit of the global and interrupt descriptor tables, respectively. The LDTR and TR registers hold the 16-bit selector for the LDT and TSS segments.

Debug and Test Registers

The 80386 includes a powerful debug feature that is built into the processor hardware. The processor provides eight registers for on-chip support of debugging operations. Four of the registers hold the linear address of the breakpoints. The processor uses one register to control the breakpoint registers while another maintains the status on each of the breakpoints. Do not use the remaining two reserved registers. Figure 12-7 illustrates the debug and test registers.

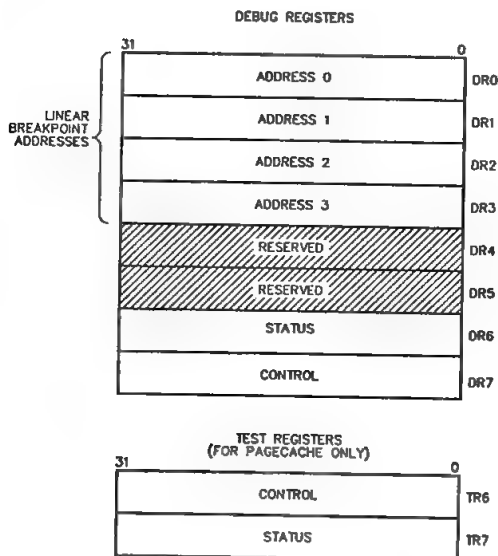


Figure 12-7. 80386 Debug and Test Registers

The 80386 also contains two registers used for testing purposes. These registers, also illustrated in Figure 12-7, control the testing of the content addressable memories in the translation lookaside buffer within the 80386. Register TR6 is the command test register and register TR7 holds the resulting data from the test.

Processor Addressing

The 80386 supports eleven different addressing modes to specify operands used in processing. Two of these modes provide for instructions that operate on register or immediate operands.

Register Operand Mode — In this mode the operand resides in one of the 8-, 16-, or 32-bit general purpose registers.

Immediate Operand Mode — In this mode the instruction itself contains the operand as part of the opcode.

The remaining nine addressing modes are the memory addressing modes. These modes allow the user to specify the effective address of the operand. The Intel documentation referred to in Chapter 4 includes a full description of these modes with examples.

The 80386 uses an advanced method of address calculation. The physical address of a given operand is based on a number of different factors. Some of the more critical factors are the effective address, the linear address, and which mode of operation the processor is executing. You can find details on the address calculation mechanism in Chapter 14.

The linear address used in these calculations consists of two components, the effective address and the segment base address. The processor calculates the effective address by summing any combination of the following four elements:

Displacement — This is an 8- or 32-bit immediate value which follows the instruction.

Base — The contents of any general purpose register makes up this value.

Index — This value may be the contents of any general purpose register except the stack register (ESP).

Scale — You can multiply the value of the index register by a specified scale factor. The permitted values for this scale factor are 1, 2, 4, or 8.

The CPU

The 80386 is capable of executing 16-bit instructions in both the real and protected modes. This feature enhances processor compatibility with the 8086 and 80286 processors. The processor accomplishes this by examining the D bit in the segment descriptor. If the value of this bit is 1, then the default operand length is 32 bits. Otherwise, the default operand length is 16 bits. Two override prefixes are available to override the value of the D bit. One prefix is the operand size prefix, the other is the address length prefix. Both of these prefixes override the value of the D bit on an individual instruction basis.

The 80386 has two distinct physical address spaces, memory address space and I/O address space. The following sections discuss these address spaces.

Memory

The 80386 views memory space as a collection of 8-bit quantities. These 8-bit collections have specific names associated with them. A group of 8-bits is a byte, 16-bits is a word, and 32-bits is a double-word (dword). The processor stores this information in memory from the lowest memory address to the highest. Thus, a word is two consecutive bytes, with the low-order byte at the lowest address. In a similar manner a dword consists of four consecutive bytes with the low-order byte also occupying the lowest address. This means that a reference to the address of a word or dword in memory is the address of the low-order byte.

The 80386 also supports two larger memory organizations: pages and segments. Segments can be of different variable lengths, shared between programs, or swapped to mass storage devices. Pages of memory consist of 4K blocks of memory.

The 80386 operates with three distinct address spaces: logical, linear, and physical. A logical address consists of a selector and an offset. (This value is also referred to as a virtual address.) An offset is essentially the same as the effective address mentioned in the previous section. It consists of a summation of the various address components (base, index, and displacement) These are combined with the scale factor to create the effective address.

The processor architecture limits each task in the 80386 to a maximum of 16K selectors. Each offset may consist of a memory block up to 4 gigabytes in size. The combination of these two factors means that a total logical address space of 64 terabytes is available per task.

Input/Output

The memory space assigned to I/O in the 80386 has a maximum size of 64K. You may arrange this area in any combination of 8-bit, 16-bit, and 32-bit ports as long as the total space does not exceed the 64K maximum. The 64K used for I/O processing resides in physical memory. I/O operations in the 80386 do not involve segmentation or paging hardware.

Any program can directly access the I/O space using the processor IN and OUT instructions. Place the desired port address in the DL, DX, or EDX register. The 80386 will zero extend all 8- and 16-bit port addresses on the upper address lines.

Interrupts

The 80386 uses two methods to alter the normal program flow: interrupts and exceptions. Exceptions handle processor instruction faults while interrupts handle asynchronous external events. This section includes a discussion of both methods.

Exceptions can be classed in one of three groups depending on the reporting method used and whether or not the processor supports instruction restart:

Faults — A class of exceptions detected and serviced before the execution of the faulting instruction. The processor supports instruction restart for this class. The return address from an exception fault routine will always point to the instruction causing the fault.

Traps — A class of exception reported immediately after instruction execution. The processor does not support instruction restart for this class. (User-defined interrupts are examples of traps.)

The CPU

Aborts — A special class of exceptions which occur when the processor cannot locate the faulting instruction. The processor does not support instruction restart for this class.

Interrupts are restricted to one of two classifications:

Maskable — A hardware interrupt occurring when the interrupt flag bit (IF) is enabled. (Disabling the IF bit turns off interrupt processing.)

Non-Maskable (NMI) — This class of hardware interrupt is reported regardless of the setting of the IF bit in the flags register. The processor provides an internal vector value of 2 for this interrupt but does not provide the usual interrupt acknowledge sequence.

Basically, all interrupts are processed in the same sequence of steps:

1. The interrupt occurs.
2. The processor saves the current program address and flag values on the stack.
3. The processor receives an 8-bit vector identifying the source of the interrupt.
4. The processor selects and executes the appropriate interrupt service routine.
5. At the completion of the service routine (IRET instruction) the processor restores the program address and flags from the stack.
6. Program execution resumes with the instruction immediately after the interrupted instruction.

The 80386 can handle a total of 256 different interrupts/exceptions. It is also possible for more than one interrupt to be in process at the same time. When a maskable interrupt occurs, the processor clears the IF flag in the flags register, disabling interrupts. If the executing program expects additional interrupt requests, the interrupt service routine must set this bit. Otherwise the processor will not recognize any further interrupt requests from the system. In the

case of a non-maskable interrupt, the 80386 will not recognize additional interrupts until it processes an IRET instruction or it is reset. If a second NMI should occur while one is in process, the 80386 will store it until the processor executes the IRET instruction.

To handle multiple interrupts, the 80386 implements a priority structure. Six different priority levels exist for interrupt processing. Table 12-5 summarizes the priority levels. If multiple interrupts occur, they will be processed according to their priority level (highest to lowest). Table 12-6 summarizes all interrupt vector assignments available in the 80386.

Table 12-5. Interrupt Priorities

PRIORITY	INTERRUPT TYPE
1	Exception faults
2	TRAP instructions
3	Debug traps
4	Debug faults
5	NMI interrupt
6	INTR interrupt

NOTE: Priority level 1 is the highest and level 6 is the lowest.

Table 12-6. 80386 Interrupt Vector Assignments

INTERRUPT VECTOR	FUNCTION
0	Divide error
1	Debug exception
2	NMI interrupt
3	One-byte interrupt
4	Interrupt on overflow
5	Array bounds check
6	Invalid OP-code
7	Device not available
8	Double fault
9	Coprocessor segment overrun
10	Invalid TSS
11	Segment not present
12	Stack fault

The CPU

Table 12-6 (continued). 80386 Interrupt Vector Assignments

INTERRUPT VECTOR	FUNCTION
13	General protection fault
14	Page fault
16	Coprocessor error
17 – 32	Reserved (do not use)
0 – 255	User available two-byte interrupt

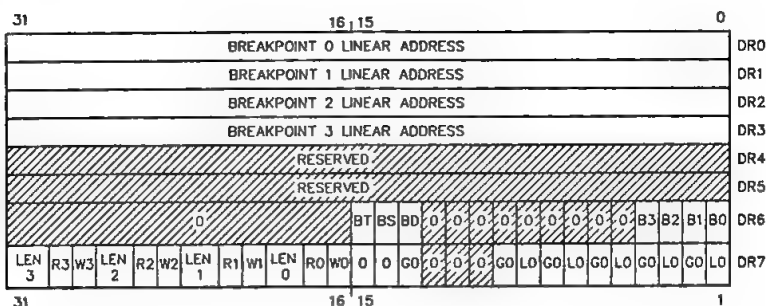
When the 80386 receives the vector identifying the interrupt source, it must determine the location of the interrupt service routine. A standard MS-DOS system can accomplish this in one of two ways.

If the processor is operating in real mode, the processor multiplies the interrupt vector by a factor of four. This multiplication provides the address of the service routine. If the processor is in protected mode, it multiplies the interrupt by a factor of 8. The resulting value is added to the base value in the interrupt descriptor table. This final value is the address of the appropriate interrupt service routine. Naturally, as with other reserved functions in the processor, reserved interrupts should not be used.

Debug Registers

The 80386 provides support, in hardware, for software debugging. These features allow the user to set breakpoints as well as single step through code. Using the debug registers, the programmer can set data access as well as code execution breakpoints. This can be especially useful for debugging code that is not accessible through more normal debug facilities.

The 80386 contains eight registers dedicated to debugging operations. Two reserved registers reduce the programmer usable total to six. Of the remaining registers, four contain breakpoint addresses and the remaining two contain control and status information. Figure 12-8 illustrates the debug registers in greater detail.



Reprinted by permission of Intel Corporation, Copyright 1987

Figure 12-8. 80386 Debug Registers

Seven bits in the DR6 register are usable by the programmer. Table 12-7 defines the function of these bits. The processor controls the setting of the flags in the DR6 register but the software control routines must clear these bits. The DR6 register is also referred to as the debug status register.

Table 12-7. DR6 Register Definitions

BIT	DEFINITION
BT	The processor sets this bit when a task switch occurs to a task where the TSS has the T bit set.
BS	The processor sets this bit if the exception handler is invoked when the TF bit in the flags register is set. This allows the debug handlers to distinguish between single step traps and other debug conditions.
BD	The processor will set this bit if the next instruction attempts to read or write the debug registers when the GD bit in DR7 is set.
B0 – B3	These bits define which of the qualified breakpoints occurred. These bits correspond to the four breakpoint registers. For example, the processor will set bit B1 if it detects breakpoint 1.

The CPU

Register DR7 is referred to as the debug control register. This register provides control over several debug features including setting breakpoints and breakpoint control options. Table 12-8 provides descriptions for the defined bit fields in register DR7.

Table 12-8. Register DR7 Bit Definitions

BIT	NAME	DEFINITION
0	L0	Local breakpoint enable 0 — This bit enables breakpoint detection for the associated breakpoint register (DR0). If this bit or the associated global bit (G0) is set, and the 80386 detects a breakpoint condition, then the exception handler is invoked.
1	G0	Global breakpoint enable 0 — This bit also enables breakpoint detection for register DR0. If set when the 80386 detects a breakpoint condition, the exception handler will be invoked.
2	L1	Local breakpoint enable 1 — Same as L0 for breakpoint register DR1.
3	G1	Global breakpoint enable 1 — Same as G0 for breakpoint register DR1.
4	L2	Local breakpoint enable 2 — Same as L0 for breakpoint register DR2.
5	G2	Global breakpoint enable 2 — Same as G0 for breakpoint register DR2.
6	L3	Local breakpoint enable 3 — Same as L0 for breakpoint register DR3.
7	G3	Global breakpoint enable 3 — Same as G0 for breakpoint register DR3.
8	LE	Local breakpoint match — If this bit or the associated GE bit is set, the 80386 will report any exact data breakpoint match immediately after the completion of the instruction that transferred the operand. This bit is automatically cleared during a task switch.
9	GE	Global breakpoint match — This bit serves the same function as the previous bit, indicating an exact data breakpoint match. This bit is unaffected by a task switch and remains in the same state.
10–12		Reserved — Do not use.

Table 12-8 (continued). Register DR7 Bit Definitions

BIT	NAME	DEFINITION															
13	GD	Global debug register access bit — If this bit is set when an instruction attempts to read or write any debug register, the 80386 will generate an exception fault. When the exception handler is invoked, it will automatically clear the exception fault. Since access to the debug registers is only permitted in real mode or in protected mode at privilege level 0, this provides extra protection against a debug register access.															
14,15		Reserved — Do not use.															
16	W0	Write memory access qualifier bit — This bit works in conjunction with the read qualifier bit to determine what type of breakpoint usage is being requested. One pair of read write qualifiers exist for each breakpoint.															
17	R0	Read memory access qualifier bit — This bit works with the write qualifier to determine breakpoint usage according to the following pattern: <table><tr><th>R</th><th>W</th><th>USAGE</th></tr><tr><td>0</td><td>0</td><td>Instruction execution only</td></tr><tr><td>0</td><td>1</td><td>Data writes only</td></tr><tr><td>1</td><td>0</td><td>Reserved</td></tr><tr><td>1</td><td>1</td><td>Data reads and writes</td></tr></table>	R	W	USAGE	0	0	Instruction execution only	0	1	Data writes only	1	0	Reserved	1	1	Data reads and writes
R	W	USAGE															
0	0	Instruction execution only															
0	1	Data writes only															
1	0	Reserved															
1	1	Data reads and writes															
18,19	LEN0	Breakpoint length bit — The 2-bit length field specifies the length of the associated breakpoint field according to the following definition: <table><tr><th>LEN</th><th>FIELD WIDTH</th></tr><tr><td>00</td><td>1 byte</td></tr><tr><td>01</td><td>2 bytes</td></tr><tr><td>10</td><td>Reserved</td></tr><tr><td>11</td><td>4 bytes</td></tr></table>	LEN	FIELD WIDTH	00	1 byte	01	2 bytes	10	Reserved	11	4 bytes					
LEN	FIELD WIDTH																
00	1 byte																
01	2 bytes																
10	Reserved																
11	4 bytes																

Each breakpoint field is aligned: 2-byte fields begin on word boundaries while 4-byte fields begin on dword boundaries.

NOTE: The read/write qualifiers and the length fields repeat for each of the remaining breakpoints. Each group requires four bits per breakpoint.

The CPU

NOTE: When using the exact data match bits (LE and GE) make sure that the handler routine requests an exact data breakpoint match. If the program does not request an exact match, a match may occur without being reported. To make sure this happens, when enabling the breakpoint bit, also enable the exact match bit.

Registers DR5 and DR4 are reserved and should not be used. Registers DR0 – DR3 are the linear breakpoint address registers. These registers contain the actual memory location of the selected breakpoints.

Test Registers

The 80386 hardware also incorporates features to allow internal testing. The 80386 can perform two major types of tests: an internal self-test and testing of the translation lookaside buffer.

The internal self-test checks about half of the internal device circuitry, including the control ROM and most of the internal non-random logic. If the test completes successfully, the 80386 will place zeroes in the EAX register. If any other value appears in the register, the device has failed the self-test. After the 80386 completes the self-test it re-initializes and begins normal operation.

Testing of the translation lookaside buffer (TLB) is more complex than the self-test. The two test registers, TR6 and TR7 store information related to this test. Figure 12-9 shows the test registers in greater detail.

31	12	11										0			
LINEAR ADDRESS			V	D	D	U	U	W	W	0	0	0	0	C	TR6
PHYSICAL ADDRESS			0	0	0	0	0	0	0	P	REP	0	0		TR7

Reprinted by permission of Intel Corporation, Copyright 1987

Figure 12-9. 80386 Test Registers

Table 12-9 provides descriptions of the bit positions used in these registers for testing purposes. TR6 is referred to as a test command register while TR7 is the test data register.

Table 12-9. Test bit Definitions

BIT	NAME	DEFINITION
Register TR6		
0	C	Command bit — This bit controls writes and lookups in the TLB. When this bit is clear (0), a write into register TR6 will cause an immediate write into the TLB. If this bit is set (1), a write to the TR6 register will cause an immediate TLB lookup.
1–4	—	Reserved bits — Do not use.
5	W	Write bit — If set, this bit allows a write from the TLB.
6	W	Write bit — If set, this bit allows writes to the TLB.
7	U	User bit — If set, this bit allows a user to read from the TLB.
8	U	User bit — If set, this bit will allow a user to write to the TLB.
9	D	Dirty bit — If set, this bit signifies that a TLB read has taken place.
10	D	Dirty bit — If set, this bit indicates a TLB write has taken place.
11	V	Valid bit — This bit is set if the current entry in the TLB was valid.
12–31		Linear address — These bits in register TR6 form the tag field for the TLB. Whenever a TLB write occurs, a TLB entry is placed in this address space. The rest of that entry is set according to the values in TR6 and TR7. When a TLB lookup occurs, the value in this memory space is compared with the TLB entry. If they match, the rest of the fields in TR6 and TR7 are set from the matching entry.
Register TR7		
0, 1	—	Reserved bits — Do Not Use.
2,3	REP	If this bit is set during a TLB write, the bits in the REP field determine which of the four TLB blocks will be written to. If the bit is clear, then the pointer in the internal paging unit makes this determination.

The CPU

Table 12-9 (continued). Test bit Definitions

BIT	NAME	DEFINITION
4	HT	Hit — This bit is used only during a TLB lookup. During a read if the bit is set (1), it indicates the lookup was a hit. A clear bit indicates a miss. During TLB writes this bit must always be set (1).
5–11		Reserved bits — Do not use.
12–31		Physical address — These bits form the data field for the TLB. When a write occurs to the TLB, the entry in the linear address space of TR6 gets set to this value. If a TLB lookup occurs, the physical address of the entry is read from this area.

Real Mode

The processor uses real mode to initialize and start operation on power-up. Generally, this mode is most often used to move the 80386 into protected mode. While in real mode, applications programs view the processor as an extremely fast 8086 with a 32-bit register set.

Real mode maintains compatibility with the 80286 by retaining the same memory size limitations, memory addressing, and interrupt handling procedures as real mode on the 80286. All 80386 program instructions are available in this mode except for instructions used in the protected model (refer to Appendix A). While in this mode the default operand size is 16-bits. Access to the 32-bit register set requires the use of override prefixes. Each memory segment has a size limit of 64K. Each 64K segment must begin on a 16 byte boundary. This implies that in this mode a 32-bit effective address cannot exceed a value of FFFFH.

The limit of available system memory in real mode is 1M. Since segments cannot exceed 64K boundaries, the 80386 will generate an exception if a data operand or data fetch operation exceeds this boundary. The processor supports segment overlapping in real mode to minimize program space requirements. Linear addresses are the same as physical addresses in this mode. The formation of the physical address is accomplished in the same manner as with the 8088 or 8086.

The processor reserves two regions of memory when operating in real mode. The interrupt table is located in the region of 00000H through 003FFH. A 4-byte jump vector is reserved for each of the 256 possible interrupts. System initialization takes place in the reserved area from FFFFFFF0H through FFFFFFFFH.

Protected Architecture

In order to adequately understand the protection mechanism employed by the 80386 CPU, you must first understand some basic concepts as they apply to the 80386. The following sections introduce these concepts and describe their implementation in the hardware of the 80386.

General Protection Concepts

One of the most powerful features of the 80386 is its ability to run in protected mode. In this mode, the CPU is capable of supporting a number of different users, each operating independently of the others. Each user interfaces with a limited portion of the computer and appears to be the only user on the system. The idea of protection is important to ensure that one user's programs and files do not interfere with any other user.

Descriptors

One of the primary responsibilities of the 80386 in protected mode is efficient memory management. The processor has to maintain discrete areas where separate user programs may be operating. To allow these areas to overlap one another would tempt disaster in the system.

One method of memory management employed by Intel processors is segmentation. Segments serve to isolate one region of memory from other regions. In the 80386 an 8-byte data structure, called a descriptor, contains information about each segment. The 80386 uses several different types of descriptors: segment, code, data, system, LDT, TSS, and gate. System descriptors are organized into tables recognized by the system hardware.

The CPU

The 80386 uses three different types of tables to store descriptor information: global, local, and interrupt. Each type of descriptor is maintained in a table for the system. A descriptor table can vary in size up to a maximum of 64K of memory. This provides enough space for 8192 8-byte descriptors per table. Each table has an associated register which holds a 32-bit linear base address and a 16-bit limit for each table.

The global descriptor table (GDT) contains the descriptors which are potentially available to all tasks in the system. The GDT can contain any type of a descriptor with the exception of interrupt descriptors. The GDT generally contains code and data segments, which are used by the operating system, and the task state segments. The first position of the GDT is not used. This position corresponds to a null selector which defines a null pointer value.

The local descriptor table (LDT) contains the descriptors associated with a specific task in the system. An LDT can only contain code, data, stack, task gate, and call gate descriptors. Normally, an operating system will provide an LDT for each task in the system. The LDT provides a means of separating the code and data segment for an individual task from the rest of the operating system. A task cannot access a segment unless the segment descriptor exists in the current LDT.

The interrupt descriptor table (IDT) contains the descriptors that point to the system interrupt service routines. The IDT can only contain task gates, interrupt gates, and trap gates. To properly service the 80386, the IDT should contain a minimum of 256 bytes of space. Each interrupt in the system must have an entry in the IDT.

Each of the above tables is associated with its own register. The register names refer to the related table; they are GTDR, LDTR, and IDTR. These registers contain the base and limit addresses for their respective tables. The information is moved to and from these registers by the appropriate 80386 instructions.

Segment Access

The 80386 permits two methods of access to the system memory: code accesses and data accesses. In order to determine if a task has legal access rights, the 80386 must be able to determine the type of segment, the instruction used, the type of descriptor, and the CPL and RPL.

Whenever an instruction loads the DS, ES, FS, or GS registers, the 80386 will perform protection validation checks. Selectors loaded into these registers must refer to data or readable code segments only. The privilege validation check consists of comparing the CPL with the EPL. If the EPL is more privileged than the CPL, a general protection fault will result. Instructions that load a selector into the stack segment must refer to the data segment descriptors to find writeable data segments.

Privilege

The 80386 employs a multiple-level protection mechanism. The four-level hierarchical privilege system employed by the 80386 is similar in many respects to the systems used on larger computer systems, such as minicomputers. In fact, the common user/supervisor mode found on many minicomputers is fully supported by the 80386 when in paging mode. The privilege mechanism is the basis for the 80386's protection scheme.

The privilege system in the 80386 provides four separate levels, 0 through 3. Level 0 is the most privileged level in this system; level 3 is the least privileged. In order to access certain system resources, the user must have an authorized privilege level equal to or greater than the level he wishes to access. This idea can be condensed into two general privilege rules that the 80386 follows:

- If data stored in a segment has a privilege level of P, then that data may only be accessed by a program with a privilege level A such that $A \geq P$.

The CPU

- If a program segment or procedure has a privilege level of P , then that program/procedure can only be called by a task with privilege level B such that $B \leq P$.

In protected mode, each task operating on the 80386 has a privilege level associated with it. The privilege level determines what resources are available to the task within the hardware and the operating system environments.

Privilege Types

A number of different privilege types are available within the 80386 system. The different types allow the 80386 to maintain the integrity of the operating system environment. The following brief discussion identifies these types.

Task Privilege — The 80386 assigns a current privilege level (CPL) to each task in the environment. The CPL can only change when the program transfers control to a code segment with a different privilege level. This transfer must occur via a gate descriptor within the system. This means that if you have an application program executing at a privilege level of 3, it can call an operating system procedure with a privilege level of 1. When this happens, the current privilege level of the application program will change to 1 until the operating system routine finishes execution.

Selector Privilege — The processor assigns a privilege level to each selector in the system. A special field, referred to as the RPL field, contains the selector's privilege level. The RPL field consists of the two least-significant bits of the selector. The processor uses the RPL field value and the CPL of the task to establish an effective privilege level (EPL). The definition of the EPL is the least-privileged level of the compared CPL and RPL fields. Changes to the RPL assignment field are made using the adjust RPL instruction. The main purpose of the RPL is to provide a means to verify that a pointer passed to an operating system procedure does not access data of a higher privilege level than the originating pointer.

I/O Privilege — The I/O privilege level (IOPL) is a special 2-bit field contained within the EFLAG register. This field defines the least-privileged level which may perform unconditional I/O instructions. In order to have unconditional access to I/O instructions, the CPL must be less than or equal to the current IOPL. The IOPL has some other tasks in the system as well.

The IOPL determines whether the CLI or STI instructions can be executed by the calling procedure. These instructions are sometimes called IOPL-sensitive. The IOPL level also determines whether or not the IF bit can be changed when loading a value in the EFLAGS register. If the level of the CPL is greater than the level of the IOPL, then the IF bit will not change when a new value loads in the EFLAGS register.

In order to speed pointer testing and verification of selector values, several instructions are provided in the CPU instruction set. The ARPL, VERR, VERW, LSL, and LAR instructions (refer to Appendix A) help to maintain system integrity by verifying that a given selector refers to the appropriate segment.

Inter-Segment Privilege Transfers

An inter-segment control transfer occurs whenever a selector is loaded into the CS register. Changing a privilege level can only happen if the operation which loaded the selector references the proper descriptor type. There are five different types of control transfers in the 80386 system:

- Intersegment within the same privilege level
- Intersegment to the same or higher privilege level
- Intersegment to a lower privilege level
- Call
- Task switch.

The CPU

A change of privilege level can only occur via a control transfer, a gate, a task switch, an interrupt, or a trap gate. If the CPL of a task changes as a result of a control transfer the 80386 will change the system stacks. The original SS:ESP values are retained in a special register called the task state segment (TSS). When a jump or call transfer executes, the new stack pointer loads into the SS and ESP registers. The previous pointer is then stored on the new stack. During a return to the original privilege level, the IRET or RET instructions restore the original stack values. If the transfer involves a subroutine, a limited number of words of data are copied from the previous stack to the current one. The number of words copied are in the gate's word count field.

Call Gates

A call gate provides the system with the ability to make protected indirect calls. The major purpose of a gate is to provide a method of making secure privilege transfers within a task. The operating system defines all gates within the system. In this way, the operating system can limit access to procedures through the gates. Gates can only be accessed by more privileged descriptors. In the same way, a gate can only transfer control to a more privileged level.

Call gates are accessed by a CALL instruction. The syntax used for this procedure is the same as a subroutine call. When a call gate is activated, the following procedures occur:

1. Load the CS:EIP from the gate and check validity.
2. Push SS value, zero-extended to 32-bits, onto stack.
3. Push ESP value onto stack.
4. Copy the 32-bit word count parameters from the old stack to the new stack.
5. Push the return address onto the stack.

NOTE: The above procedures are the same as on a 80286 based system except for the use of 32-bit values instead of 16-bit values.

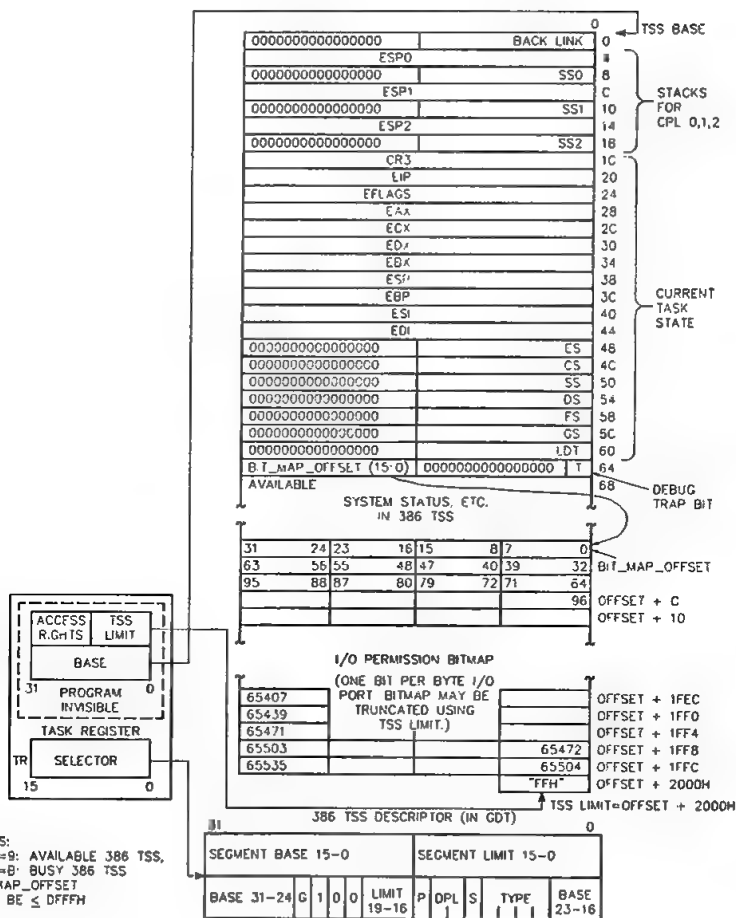
Task Switching

Task switching is an essential capability in a multi-tasking/multi-user operating system. This capability allows the operating system to switch between different procedures. The faster the operating system can accomplish this, the better.

The 80386 enhances this level of performance by providing a special task switch instruction within the hardware. When a task switch executes, the 80386 will save the entire machine state, load a new execution state, and perform protection checks. This entire procedure is accomplished in approximately 17 microseconds.

A program can invoke the task switch operation by using an inter-segment JMP or CALL instruction. The instruction must reference a task state segment (TSS) or a task gate descriptor within the GDT or the LDT. The task switch operation can also be invoked by an exception, trap, or external interrupt, provided there is a task gate descriptor in the IDT descriptor slot. Figure 12-10 illustrates the 80386 task state segment format.

A TSS descriptor will point to an entire segment of memory containing the current 80386 execution state. A task gate descriptor, on the other hand, only contains a TSS selector. The TSS for the 80386 must contain a minimum of 64K of space. Any free space in the TSS is available for the operating system to use for the storage of additional information. For the sake of compatibility, the 80386 will also operate with an 80286-style TSS.



Reprinted by permission of Intel Corporation, Copyright 1987

Figure 12-10. 80386 TSS and TSS Registers

The 80386 requires that each task within the system be associated with a TSS. The task state segment register (TR) is a special register in the 80386 that identifies the current TSS. The TR contains a selector which refers to a descriptor that defines the current TSS. Whenever the TR loads a new selector, a hidden associated base and limit register, illustrated in Figure 12-11, is also loaded. The IRET instruction, if properly set, will return control to the interrupted task.

Part of the information the operating system requires about the task state resides in the flag register and the machine status word. Bit 14 in the EFLAGS register (the NT bit) controls the operation of the IRET instruction. If this bit is set to a 1, the IRET instruction will perform a task switch return, restoring control to the proper task. If this bit is set to a 0, IRET will perform a normal return. The machine status register (CR0) contains information dealing with the state of the coprocessor during a task switch.

Setting or resetting the NT bit requires a specific procedure. A CALL or INT instruction initiates the task switch operation. The 80386 will then mark the new TSS as busy and establish a back link field in the new TSS to the old TSS selector. The TSS changes to busy by changing the descriptor field type from 9H to BH. The NT bit in the new TSS is set by CALL- or INT-initiated task switches. If an interrupt does not cause a task switch, it will clear the NT bit. The 80386 restores the NT bit on completion of the interrupt handler execution.

The 80386 does not automatically save the current state of the coprocessor during a task switch. The reason for this is that the requirements of the incoming task are unknown. The incoming task may not even use the coprocessor. The 80386 handles this situation by setting the TS bit in CR0 each time a task switch occurs. The 80386 will then detect the first use of a coprocessor instruction after the task switch. At that point the 80386 will issue a processor extension exception. It is then the responsibility of the exception handler to determine whether or not to save the state of the coprocessor.

The CPU

Paging

Paging is one additional method of system memory management employed by the 80386. Like the other methods already mentioned, paging has applications in the multi-tasking operating system environment. While segmentation methods tend to modularize programs or data, paging divides a program into multiple, uniformly sized pages. Since most programs exhibit a locality of reference, this is advantageous to the programmer. Because of this, only a small number of active pages from each task need to be active in memory at any given time. Each page frame in the 80386 environment consists of 4096 bytes of memory.

Page Descriptor Base Register — Register CR3 is the page directory physical base address register. This register contains the physical starting address of the page directory. The lower 12 bits all contain zeros to ensure that the page directory remains page aligned. If a program loads this register by using the MOV CR3,reg instruction, the page table entry cache will be flushed.

Page Directory — Each page directory entry is four bytes long. The directory itself contains 4096 bytes, providing enough room for 1024 entries. Each entry contains the address of the next level of tables, the page tables, and related information. Figure 12-11 illustrates the format of a typical page directory entry. Table 12-10 describes the table entries themselves.

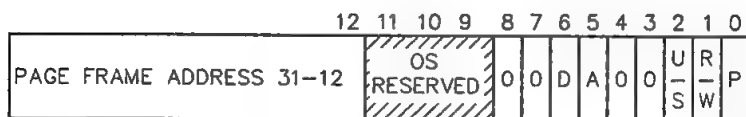


Figure 12-11. Page Directory Entry Format

Table 12-10. Page Directory/Page Table Entries

BIT	NAME	DESCRIPTION																				
0	P	Present. This bit indicates whether or not an entry can be used for address translation. If the bit is set (1), it can be used, otherwise it cannot. If the entry is not used for address translation, the remaining bits of the entry space become available to the system.																				
1	R/W	Read/Write. This bit, in conjunction with the U/S bit (bit 2), provides a protection attribute for each page entry reference. The setting of the R/W and U/S bits determines the access level permitted for the page. The following table summarizes the permitted page access levels.																				
		<table><tr><th>U/S</th><th>R/W</th><th>LEVEL 3</th><th>LEVELS 0, 1, OR 2</th></tr><tr><td>0</td><td>0</td><td>None</td><td>Read/Write</td></tr><tr><td>0</td><td>1</td><td>None</td><td>Read/Write</td></tr><tr><td>1</td><td>0</td><td>Read-Only</td><td>Read/Write</td></tr><tr><td>1</td><td>1</td><td>Read/Write</td><td>Read/Write</td></tr></table>	U/S	R/W	LEVEL 3	LEVELS 0, 1, OR 2	0	0	None	Read/Write	0	1	None	Read/Write	1	0	Read-Only	Read/Write	1	1	Read/Write	Read/Write
U/S	R/W	LEVEL 3	LEVELS 0, 1, OR 2																			
0	0	None	Read/Write																			
0	1	None	Read/Write																			
1	0	Read-Only	Read/Write																			
1	1	Read/Write	Read/Write																			
2	U/S	User/Supervisor. Refer to the description of bit 1.																				
3,4	—	Reserved. Do not use these bits.																				
5	A	Accessed. The 80386 sets this bit prior to read or write accesses to the referenced entry. If a software program modifies this bit, it should use the LOCK prefix to make sure the page retains its integrity.																				
6	D	Dirty. The 80386 sets this bit prior to a write to the referenced page table entry. This bit is undefined for page directory entries. If a software program modifies this bit, it should use the LOCK prefix to make sure the page retains its integrity.																				
7,8	—	Reserved. Do not use these bits.																				

The CPU

Table 12-10 (continued). Page Directory/Page Table Entries

BIT	NAME	DESCRIPTION
9-11	—	OS reserved. These bits are reserved for use by the operating system. One example would be to use this area to record page aging. By knowing how long a page has resided in memory, a least recently used replacement algorithm could be implemented.
12-31	—	These bits are where address information for the entry is stored. For a page directory, this area is the page table address. For a page table, the page frame address is available here.

Page Table — The page table is the next level of information below the page directory. The page table also contains 4096 bytes of data space. This provides sufficient room for 1024 table entries since each entry consist of four bytes. The format for the page table entry is identical to the page directory entry format. The page table entry contains the starting address of the page frame and information concerning the page.

Translation Lookaside Buffer

The translation lookaside buffer (TLB) is a hardware implemented, four-way, set associative, 32-entry, page table cache. This feature was added to the 80386 as an aid to support demand paged virtual memory systems. The cache helps to prevent performance degradation by removing the requirement that the 80386 access two levels of tables for each memory reference. Instead, the TLB maintains information on the most recently accessed page table entries. Since the TLB can hold 32 entries, and each page contains 4K, the TLB can cover 128K of memory addresses. This will normally translate into a 98% hit rate in the average multi-tasking system. The TLB operates in conjunction with the 80386's paging mechanism.

Whenever the paging hardware is active, the segmentation unit passes a 32-bit linear address to the paging unit. The paging unit compares the upper 20 bits with all entries in the TLB. If one entry matches, referred to as a TLB hit, then the 32-bit physical address is calculated and placed on the address bus.

If, on the other hand, the entry is not in the TLB, then the 80386 reads the appropriate page directory entry. If the P bit in the page directory entry is set (1), it indicates that the appropriate page table is in memory. The 80386 then reads the entry and sets the access bit. If the P bit is set (1) in the page table entry, indicating that it is also in memory, then the 80386 will update the access and dirty bits and fetch the operand. The 80386 then stores the upper 20 bits of the linear address in the TLB for future access.

If the P bit is clear (0) in the page table entry or the page directory entry, then the processor generates a page fault. A page fault could also be generated if a memory reference violated the page protection attributes. If a page fault occurs, the CS:EIP register will point to the instruction causing the error and a 16-bit error code will be placed on the stack by the page fault handler. The operating system uses the error code to try to determine how to handle the page fault error.

Only three bits in the error code are used to provide information concerning the error. Table 12-11 describes these bits and the functions that they perform.

Table 12-11. Page Fault Error Code Bits

BIT	NAME	FUNCTION
0	P	Page present. If this bit is set (1), the error was caused by a page protection violation. If the bit is clear, the error resulted from attempted access to a not present page.
1	W/R	Write/Read. If the bit is set, the access that caused the error was a write. Otherwise the error resulted from a read access. These bits work in conjunction with the U/S bit.

The CPU

Table 12-11 (continued). Page Fault Error Code Bits

BIT	NAME	FUNCTION															
2	U/S	User/Supervisor. If the bit is set, the processor was in user mode when the access fault occurred. If the bit is clear, then it was in supervisor mode. The following table correlates the bit pattern with the function.															
<table> <tr> <th>U/S</th><th>W/R</th><th>TYPE OF ACCESS</th></tr> <tr> <td>0</td><td>0</td><td>Supervisor Read</td></tr> <tr> <td>0</td><td>1</td><td>Supervisor Write</td></tr> <tr> <td>1</td><td>0</td><td>User read</td></tr> <tr> <td>1</td><td>1</td><td>User write</td></tr> </table>			U/S	W/R	TYPE OF ACCESS	0	0	Supervisor Read	0	1	Supervisor Write	1	0	User read	1	1	User write
U/S	W/R	TYPE OF ACCESS															
0	0	Supervisor Read															
0	1	Supervisor Write															
1	0	User read															
1	1	User write															

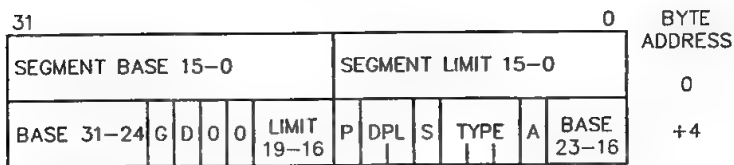
NOTE: The remaining bits in the 16-bit error code are undefined and should not be used.

Descriptor Attributes

Each descriptor in the 80386 consists of eight bytes. The descriptor contains the attributes which describe the segment in memory. The attributes contained in the descriptor are:

- The 32-bit base linear address of the segment,
- The 20-bit segment length,
- A 1-bit granularity field,
- The segment protection level,
- The segment read, write, and execute privileges,
- The segment default operand size,
- The segment type.

Figure 12-12 illustrates the general layout of the segment descriptor.



Reprinted by permission of Intel Corporation, Copyright 1987

Figure 12-12. Segment Descriptor Format

The first two bytes of the descriptor are the first 16 bits (0 – 15) of the 20-bit segment length. The remaining four bits (16 – 19) are the first four bits of the seventh byte of the descriptor. In a similar fashion the 32-bit segment address information resides within several bytes in the descriptor. The second and third bytes of the descriptor contain the first 16 bits (0 – 15) of the segment address. Bits 16 through 23 of the segment address are in the fifth byte of the descriptor. The remaining eight bits (24 – 31) are in the last byte of the descriptor. Each of the descriptor types within the 80386 vary this general format to meet the requirements of their specific descriptor class.

Whenever a descriptor is accessed by the 80386, the A (access) bit is set. The G bit is the granularity bit. This bit specifies whether a segment is byte-granular or page-granular. If the granularity bit is set (1), the segment is page-granular. The maximum size of a page-granular segment is four gigabytes with each page consisting of 4K. If the bit is clear (0), the segment is byte-granular. The maximum size of this type of segment is one megabyte.

The D bit determines the default length for operands and effective addresses. If this bit is set (1), the default length is 32-bits. If the bit is clear (0), the default length is 16-bits.

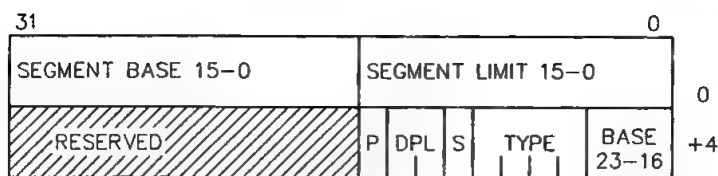
The DPL field is the individual descriptor privilege level field. These two bytes describe the descriptor privilege level from 0 – 3. The P bit is the present bit. If this bit is set (1), the segment exists in physical memory. If a program attempts to access a segment that is not present, the processor will generate a not present exception.

The CPU

The 80386 recognizes two categories of segments: system and non-system. A system segment contains information pertaining to the operating system. A non-system segment contains code and data only. The system uses the S bit to determine what type of segment it is dealing with. If the S bit is set (1), then the segment is either a code or a data segment. A cleared bit indicates a system segment.

Code and Data Descriptors

Code and data descriptors follow the same general format as illustrated in Figure 12-12. The primary difference between the two formats is the use of an access rights byte in the code and data descriptors. The access rights byte contains information describing the segment privilege level, access mode, read and write privileges, and whether or not the segment is code or data. Figure 12-13 is an illustration of the descriptor format for the code and data descriptors. Table 12-12 describes the bit definitions used in the access rights byte of the descriptor.



Reprinted by permission of Intel Corporation, Copyright 1987

Figure 12-13. Code and Data Descriptor Format

Table 12-12. Access Rights Byte Bit Definitions

BITS	NAME	DESCRIPTION
0	A	Accessed. The system uses this bit to determine whether or not a segment is present in memory. If the bit is set (1), then the segment is present in memory. If the bit is clear (0), the segment has not been accessed.

Table 12-12 (continued). Access Rights Byte Bit Definitions

BIT	NAME	DESCRIPTION
CODE SEGMENT (see note)		
1	R	Readable. If this bit is set, the code segment may be read by the program. If the bit is clear, it can not.
2	C	Conforming. When this bit is set, the code segment may only be executed when the CPL is greater than or equal to the DPL. The CPL must remain unchanged. Segments that conform may be executed by programs with different privilege levels.
3	E	Executable. The system uses this bit to determine if the segment is a code segment or a data segment. If the bit is clear, it is a code segment.
DATA SEGMENT (see note)		
1	W	Writeable. If this bit is set, the data segment may be written to. If the bit is clear, it can not.
2	ED	Expansion direction. This bit determines in which direction the segment will expand in memory. If the bit is clear the segment will expand upward in memory and offsets must be less than or equal to the limit. (Upward expansion indicates a data segment.) If the bit is set, the segment will expand downward in memory. (Downward expansion indicates a stack segment) In this case the offsets must be greater than the limit.
3	E	Executable. The system uses this bit to determine if the segment is a code segment or a data segment. If the bit is clear, it is a data segment.
4	S	Segment descriptor. The system uses this bit to determine if the descriptor is code, data, or system. If this bit is set, the descriptor is a code or a data descriptor. If the bit is clear, the descriptor is a system or a gate descriptor.

The CPU

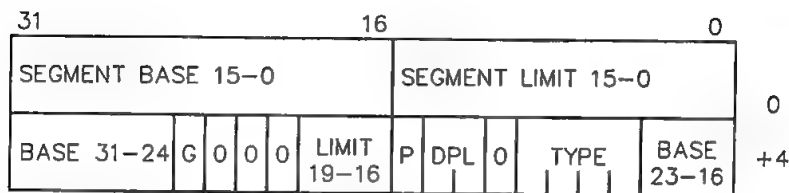
Table 12-12 (continued). Access Rights Byte Bit Definitions

BITS	NAME	DESCRIPTION
5-6	DPL	Descriptor privilege level. These two bits determine the privilege level of the descriptor.
7	P	Present. This bit determines if the segment is mapped into the physical memory. If the bit is set, the segment exists in memory.

NOTE: These three bits have different functions in the code descriptor and the data descriptor. The remaining bits function the same with either type.

System Segment Descriptors

The 80386 recognizes 12 different types of system segment descriptors. These descriptors provide information about operating system tables, tasks, and gates. Figure 12-14 illustrates the general form of the system segment descriptor. Table 12-13 lists the types of system segments that may appear in the descriptor type block.



Reprinted by permission of Intel Corporation, Copyright 1987

Figure 12-14. System Segment Descriptor Format

Table 12-13. System Segment Type

TYPE	SEGMENT DEFINED AS
0000	Invalid
0001	Available 286 TSS
0010	LDT
0011	Busy 286 TSS
0100	286 call gate
0101	Task gate (286 or 386)
0110	286 interrupt gate
0111	286 trap gate
1000	Invalid
1001	Available 386 TSS
1010	Undefined
1011	Busy 386 TSS
1100	386 call gate
1101	Undefined
1110	386 interrupt gate
1111	386 trap gate

A 80386 system descriptor contains a 32-bit base linear address and a 20-bit segment limit. This differs from the 80286 system descriptor which contains a 24-bit base address and a 16-bit limit. This means that an 80386 descriptor can be easily identified. The upper 16-bits of the descriptor will all be zeros.

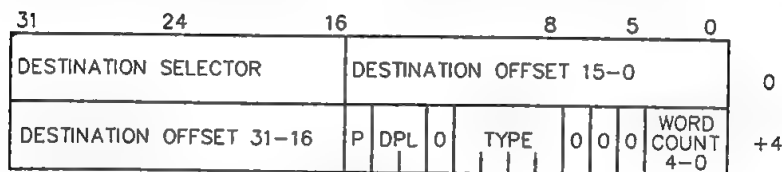
The following material describes the different classes of system segment descriptors.

LDT — The LDT descriptor contains information about the local descriptor tables. These descriptors are only allowed in the global descriptor table. The DPL field is ignored due to the fact that the load instruction is only available in privilege level 0.

TSS — This descriptor contains information about the location, size, and privilege level of a task state segment. The TSS itself is a special segment that contains all state information for a task. It also contains a linkage field to permit the nesting of tasks.

The CPU

Gate — Gates control access to entry points within the target code segment. There are four different types of gates: call, task, trap, and interrupt. This mixture allows the programmer to control the entry points to the operating system. Call gates are used to change privilege levels, task gates to perform a task switch, and interrupt and trap gates to specify interrupt service routines. Gate descriptors use a slightly modified format, as illustrated in Figure 12-15.



Reprinted by permission of Intel Corporation, Copyright 1987

Figure 12-15. Gate Descriptor Format

A program generally uses a call gate to transfer control to a higher privilege level. The descriptor used in this process consists of three distinct fields. The three fields are an access byte, a long pointer (consisting of a destination selector and offset), and a word count. The pointer points to the start of a routine and the word count indicates how many parameters to copy from the calling program stack to the called routine's stack. The word count field is only used if there is a change of privilege level; otherwise it is ignored.

A task gate only uses the selector portion of the descriptor. Since task gates are only used to switch tasks, they may only refer to the TSS. That is why only the selector is used.

Trap gates handle the selector and offset in a different manner. In a trap gate the selector and offset form a pointer. The pointer indicates the beginning of the interrupt or trap service routines. The main difference between a trap gate and an interrupt gate is that the interrupt gate disables interrupts. The trap gate does not disable interrupts.

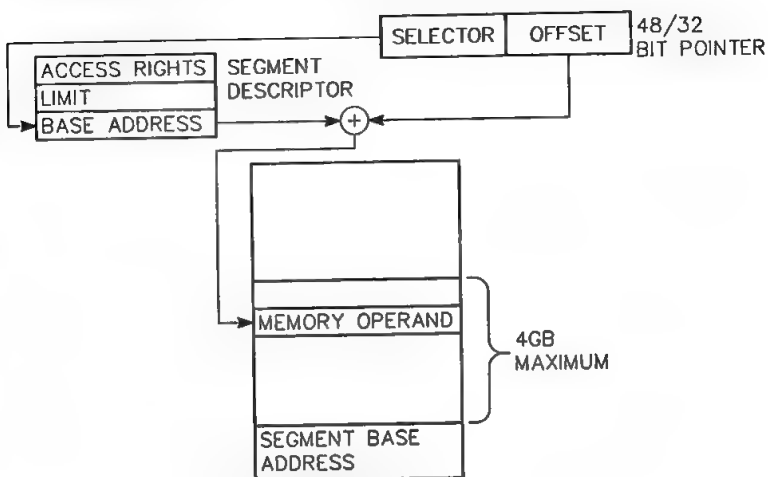
To improve processing throughput, a special system descriptor cache is provided within the 80386. Each segment register has an associated segment descriptor cache register. Whenever a segment

register's contents change, the 8-byte descriptor associated with the selector is cached. Thereafter, each reference to that segment comes from the cache instead of the descriptor. The contents of the descriptor cache is not accessible to the programmer.

NOTE: The contents of the descriptor cache only changes when a segment register is changed. If a program modifies a descriptor table, it must also reload the appropriate registers after changing the descriptor's value.

Memory Addressing

In protected mode, the 80386 uses a method similar to that used in real mode to determine physical addresses. The system uses a 16-bit selector to determine the linear base address of the segment. The base address is then added to a 32-bit effective address to form a 32-bit linear address. If the paging mode is enabled, the paging unit maps the 32-bit linear address into the memory space. Otherwise, the 32-bit linear address becomes the 32-bit physical address. The difference between real mode and protected mode is in how the base address is determined. Figure 12-16 illustrates the addressing mechanism used in protected mode.



Reprinted by permission of Intel Corporation, Copyright 1987

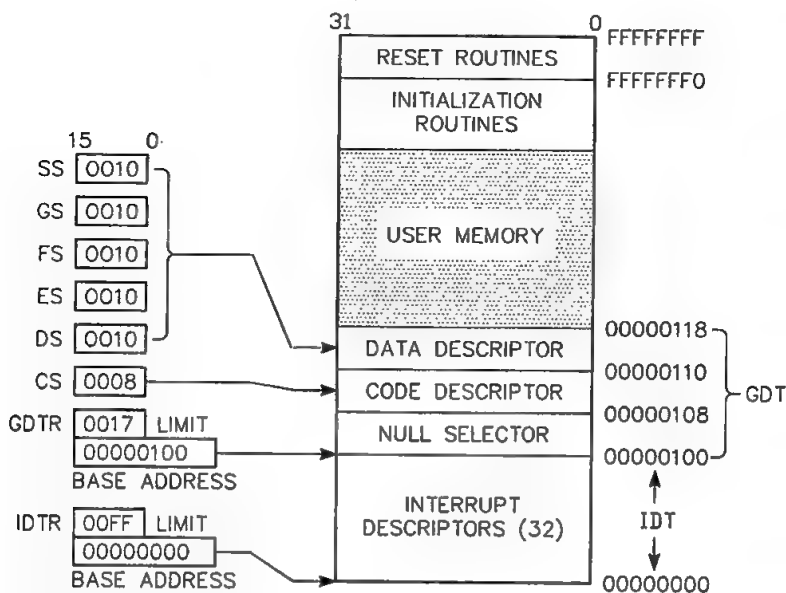
Figure 12-16. Protected Mode Addressing

The CPU

While in protected mode, the selector specifies an index into a table defined by the operating system. The table contains the 32-bit base address of the desired segment. The physical address is then formed by adding the base address contained in the table to the offset.

Protected Mode Initialization

The 80386 always begins operation immediately after a reset in real mode. Because of this, certain registers must be initialized in order to change into protected mode. Figure 12-17 illustrates a simple protected system while Figure 12-18 shows the descriptors for this system.



Reprinted by permission of Intel Corporation, Copyright 1987

Figure 12-17. Protected System Model

The CPU

Using this method the GDT should contain two TSS descriptors in addition to the code and data data descriptors required for the first task. The TSS register must be initialized to point to a valid TSS descriptor.

Virtual 8086 Mode

Virtual 8086 mode is an extension of the 80386's protected mode environment. In this mode the 80386 permits the execution of 8086-based programs as in real mode, but with more flexibility. While in this mode it is possible to run an 8086 application and operating system, an 80386 operating system, and an 80386 application with its operating system simultaneously. In a multi-user environment, this can provide a great deal of responsiveness to user needs.

Virtual Mode Addressing

In virtual mode, memory address calculations are performed in the same manner as in real mode. The contents of the segment register is added to the offset after first being shifted left four bits. The resulting value is the base linear address.

The 80386 also allows the operating system to determine, on a per task basis, how a program will access memory. Any program can access memory by using the 8086 address mechanism or the protected mode addressing mechanism.

In virtual mode each program is allowed access to a memory area of 1 megabyte. This memory area can be mapped anywhere within the allowable address space of the 80386. In virtual mode, as in real mode, if a program's effective address exceeds 64K, the 80386 will generate an exception.

Paging

It is not required to activate the paging hardware while operating in the virtual mode. However, it is recommended, since paging does provide some protection and operating system isolation. Paging

should be used whenever there are multiple virtual mode tasks executing. Paging must be used to relocate the address space of a virtual mode task above the 1 megabyte boundary.

The paging hardware within the 80386 permits a 20-bit linear address, produced by a virtual mode program, to be divided into 256 distinct pages. Each page may be located anywhere within the allowable address space of the 80386. Since the page directory base register (CR3) is loaded by a task switch, a virtual mode task can use different mapping schemes. The paging hardware also allows multiple applications to share 8086 operating system code.

Protection and I/O Permission

Each program running in virtual mode is assigned a privilege level of 3. If a program attempts to execute a privileged instruction while in virtual mode, the processor will generate an exception. This differs from real mode since all programs in real mode execute at a privilege level of zero. The following instructions are considered privileged and may only be executed at privilege level 0:

LIDT;	MOVDRn,reg;	MOVreg,DRn
LGDT;	MOVTRn,reg;	MOVreg,TRn
LMSW;	MOVCRn,reg;	MOVreg,CRn
CLTS;		
HLT;		

The following instructions are only available in protected mode:

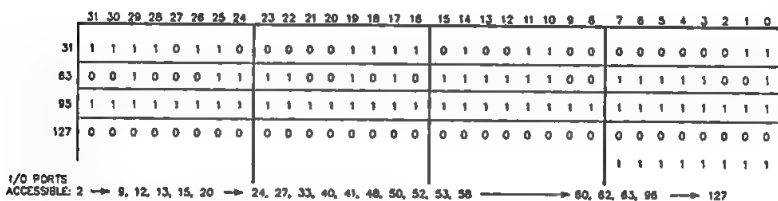
LTR;	LLDT;	LAR;	LSL;
ARPL;	STR;	SLDT;	VERR;
VERW;			

The following instructions are IOPL-sensitive in protected mode:

IN;	OUT;	INS;	OUTS;
REP INS;	REP OUTS;	STI;	CLI;

INT n;	PUSHF;	POPF;	STI;
CLI;	IRET;		

The I/O permission bit map can be thought of as a bit string, ranging in size from 0 – 64 kilobits. The string is located in the current TSS at an offset referred to as bit map offset. The location of this offset must be such that the bit string does not exceed a value of DFFFH. This way, the entire bit map, and the byte which follows it, can be located in a space that is offset less than FFFFH from the TSS base. (Refer to the illustration of the system TSS in Figure 12-10.) Figure 12-19 illustrates an example of a typical I/O permission bit map found in the TSS.



Reprinted by permission of Intel Corporation, Copyright 1987

Figure 12-19. I/O Permission Bit Map

Each bit in the I/O permission bit map corresponds with an 8-bit system I/O port. If the bit for the respective port is set to 0, the system will permit I/O with that port. Since each I/O port must be protectable, all bits corresponding to a word or dword port must be set to zero before I/O will be permitted with that port. If they are not, the processor will generate an exception error.

The bit map can generally be located anywhere within the TSS provided the guideline values stated earlier are observed. If the bit map is placed beyond the limit of the TSS segment, in other words ignored, then an additional 8K of memory normally used for the bit map would be freed for the system.

NOTE: The last byte of information in the I/O permission bit map must contain all 1s. This byte must exist within the limit of the defined TSS segment.

Interrupts

Interrupt handling in virtual 8086 mode is more complicated than in the real or protected modes discussed earlier. In virtual 8086 mode all generated interrupts and exceptions require a privilege change back to the 80386 operating system level. The 80386 operating system then handles the interrupt or exception and returns control to the 8086 program.

The 80386 operating system can determine whether the interrupt came from a protected mode or a virtual mode program by examining the VM bit in the EFLAGS register. If a virtual mode program is interrupted, this bit will be cleared (0). However, in either case, the VM bit in the EFLAG image on the stack will remain unchanged.

Using this interrupt method, the 80386 can provide a totally transparent 8086 environment for an 8086 application. It does this by intercepting, and then emulating, 8086 operating system calls and IN and OUT instructions.

Virtual Mode Transitions

There are basically two available methods used by the 80386 to enter virtual 8086 mode. The first method requires that the 80386 already be in protected mode. To enter virtual mode by this method, switch to a task with a 386 TSS that has the VM bit in the EFLAGS image set to 1. To enter using the second method, execute a 32-bit IRET instruction at privilege level 0 with the VM bit set to 1 in the EFLAGS image.

The instruction set of the 80386 makes it appear that other methods of entry are possible, but that is not the case. For example, the POPF instruction does not affect the VM bit even if the processor is already in protected mode or privilege level 0. PUSHF, on the other hand, always pushes a 0 in the VM bit, even if the processor is already in virtual 8086 mode. Once it does that, it is impossible for the program to tell if it is executing in real or virtual mode.

The transition out of virtual 8086 mode returns to the 80386 protected mode. This can occur only when the processor receives an interrupt or an exception while executing in virtual 8086 mode. In virtual 8086 mode all interrupts and exceptions are vectored through the protected mode IDT and end up in an interrupt handler in the protected mode. This means that performing a task switch or an interrupt will exit virtual 8086 mode. During this process the VM bit is cleared.

If an interrupt or a trap gate is used to handle an interrupt or exception from virtual 8086 mode, a matching IRET must be executed at privilege level 0. The gate must perform an inter-level interrupt only to level 0.

The following material provides a general sequence of steps to illustrate the method used to exit virtual 8086 mode via an interrupt and to return to the interrupted program. The first sequence of steps covers the exit procedure to the interrupt handler while the second covers the return to the program.

To exit from virtual 8086 mode via interrupt gate:

1. Save the FLAGS register and turn off the VM and TF bits. If the interrupt is serviced by an interrupt gate, turn off the IF bit.
2. Switch from the virtual stack to the TSS level 0 stack. Save the virtual 8086 mode SS and ESP registers.
3. Push the 8086 register values onto the new stack in the following order: GS, FS, DS, ES. Push these values as 32-bit quantities with the upper 16-bits undefined. Load all four registers with null selectors (0).
4. Push the old 8086 stack pointer onto the new stack (push it as 32-bits with the high 16-bits undefined), and then push the 32-bit EIP register saved earlier.
5. Push the 32-bit flag register saved in step 1.
6. Push the old 8086 instruction pointer onto the new stack. Do this by pushing the CS register (32-bits, high 16 bits undefined), followed by the 32-bit EIP register.
7. Load the new CS:EIP value from the interrupt gate and begin execution of the interrupt routine in protected mode.

To return to the interrupted program:

1. First, check the value of the NT bit in the FLAGS register. If this bit is on, an interrupt task is performed. The current state is stored in the current TSS and the link field in the current TSS is used to locate the TSS for the interrupted task you wish to resume. Otherwise, continue as follows:
2. Read the value of the FLAGS image from SS:8[ESP] into the FLAGS register. (This restores the VM bit.)

The CPU

3. Pop the instruction pointer CS:EIP. Pop EIP first, and then the 32-bit word containing CS. CS is contained in the lower 16 bits of this word. (If VM=0, this is a protected mode segment load; if VM=1, it is an 8086 segment load.)
4. Increment the ESP register by 4. (This bypasses the FLAGS image from step 1.)
5. If VM=1, load registers ES, DS, FS, and GS respectively from memory locations SS:[ESP+8], SS:[ESP+12], SS:[ESP+16], and SS:[ESP+20]. (These are 8086 segment loads.) Otherwise, check that the ES, DS, FS, and GS selectors are valid in the interrupted routine. Null out any invalid selectors.
6. If $RPL(CS) > CPL$, pop the stack pointer SS:ESP from the stack. Pop the ESP register first, then the 32-bits containing SS. The lower 16-bits contain the value of SS. (If VM=1, load as a protected mode segment register load, if VM=0, load as an 8086 segment register.)
7. Resume execution of the interrupted program. (The value of the VM bit determines whether the processor resumes in protected mode, or virtual 8086 mode.)

80386 Pinouts

Figure 12-20 illustrates the locations of the various signal pins on the 80386 processor package. The figure includes both a top and a bottom view of the processor. This pinout is referred to as a pin grid array (PGA) package and is used because of the high signal density of this device.

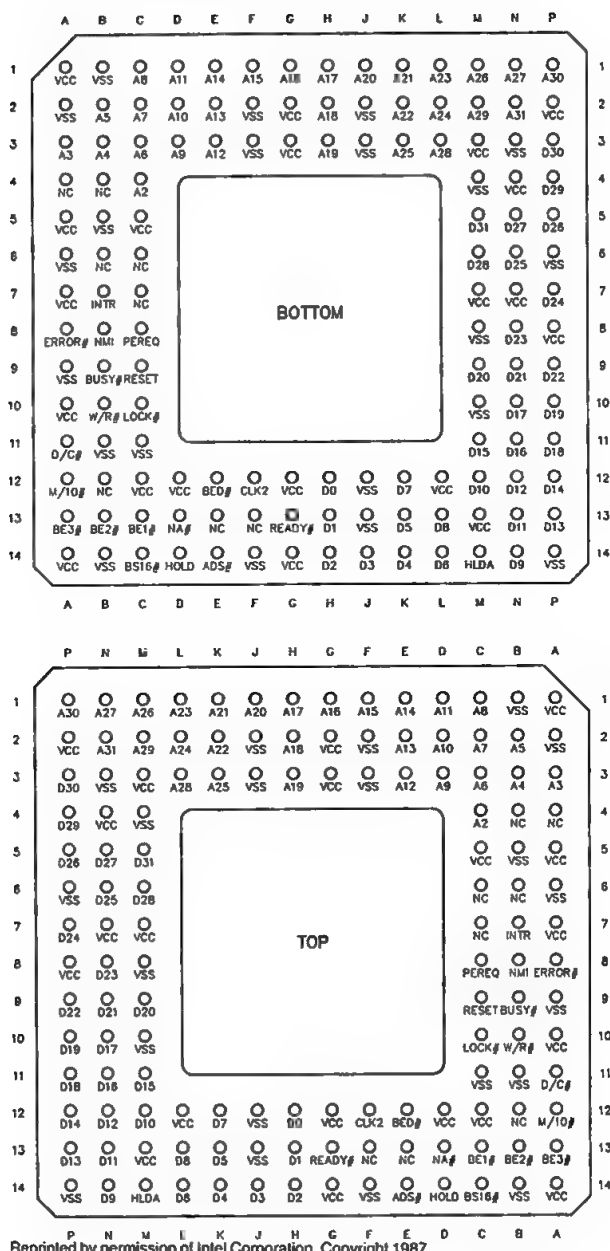


Figure 12-20. 80386 Pin Assignments

The CPU

Tables 12-14 and 12-15 list the signals available within the 80386. Table 12-14 lists the signals in alphabetical order. Table 12-15, on the other hand, lists the signals by the pin numbers in Figure 12-20. You can locate a specific pin by intersecting the row and the column guide numbers in column and row order.

Table 12-14. Device Pinout Grouped by Function

SIGNAL	PIN	SIGNAL	PIN	SIGNAL	PIN	SIGNAL	PIN
A31	N2	D31	M5	RESET	C9	VCC	A1
A30	P1	D30	P3	HOLD	D14	VCC	A5
A29	M2	D29	P4	HLDA	M14	VCC	A7
A28	L3	D28	M6	CLK2	F12	VCC	A10
A27	N1	D27	N5	$\overline{\text{ADS}}$	E14	VCC	A14
A26	M1	D26	P5	$\text{W}/\overline{\text{R}}$	B10	VCC	C5
A25	K3	D25	N6	$\text{D}/\overline{\text{C}}$	A11	VCC	C12
A24	L2	D24	P7	$\text{M}/\overline{\text{IO}}$	A12	VCC	D12
A23	L1	D23	N8	LOCK	C10	VCC	G2
A22	K2	D22	P9	$\overline{\text{NA}}$	D13	VCC	G3
A21	K1	D21	N9	$\overline{\text{BS16}}$	C14	VCC	G12
A20	J1	D20	M9	READY	G13	VCC	G14
A19	H3	D19	P10	INTR	B7	VCC	L12
A18	H2	D18	P11	PEREQ	C8	VCC	M3
A17	H1	D17	N10	$\overline{\text{BUSY}}$	B9	VCC	M7
A16	G1	D16	N11	ERROR	A8	VCC	M13
A15	F1	D15	M11	NMI	B8	VCC	N4
A14	E1	D14	P12	N.C.	A4	VCC	N7
A13	E2	D13	P13	N.C.	B6	VCC	P2
A12	E3	D12	N12	N.C.	B12	VCC	P8
A11	D1	D11	N13	N.C.	C6	GND	A2
A10	D2	D10	M12	N.C.	C7	GND	A6
A9	D3	D9	N14	N.C.	E13	GND	A9
A8	C1	D8	L13	N.C.	F13	GND	B1
A7	C2	D7	K12	GND	N3	GND	B5
A6	C3	D6	L14	GND	B11	GND	B14
A5	B2	D5	K13	GND	C11	GND	F2
A4	B3	D4	K14	GND	F3	GND	F3
A3	A3	D3	J14	GND	F14	GND	J2
A2	C4	D2	H14	GND	J3	GND	J12
$\overline{\text{BE3}}$	A13	D1	H13	GND	J13	GND	M4
$\overline{\text{BE2}}$	B13	D0	H12	GND	M8	GND	M10
$\overline{\text{BE1}}$	C13	$\overline{\text{BE0}}$	E12	GND	P6	GND	P14

Table 12-15. Device Pinout Grouped by Function

SIGNAL PIN		SIGNAL PIN		SIGNAL PIN		SIGNAL PIN	
A1	VCC	C6	N.C.	H1	A17	M10	GND
A2	GND	C7	N.C.	H2	A18	M11	D15
A3	A3	C8	PEREQ	H3	A19	M12	D10
A4	N.C.	C9	RESET	H12	D0	M13	VCC
A5	VCC	C10	LOCK	H13	D1	M14	HLDA
A6	GND	C11	GND	H14	D2	N1	A27
A7	VCC	C12	VCC	J1	A20	N2	A31
A8	ERROR	C13	BE1	J2	GND	N3	GND
A9	GND	C14	BS16	J3	GND	N4	VCC
A10	VCC	D1	A11	J12	GND	N5	D27
A11	D/C	D2	A10	J13	GND	N6	D25
A12	M/IO	D3	A9	J14	D3	N7	VCC
A13	BE3	D12	VCC	K1	A21	N8	D23
A14	VCC	D13	NA	K2	A22	N9	D21
B1	GND	D14	HOLD	K3	A25	N10	D17
B2	A5	E1	A14	K12	D7	N11	D16
B3	A4	E2	A13	K13	D5	N12	D12
B4	N.C.	E3	A12	K14	D4	N13	D11
B5	GND	E12	BE0	L1	A23	N14	D9
B6	N.C.	E13	N.C.	L2	A24	P1	A30
B7	INTR	E14	ADS	L3	A28	P2	VCC
B8	NMI	F1	A15	L12	VCC	P3	D30
B9	BUSY	F2	GND	L13	D8	P4	D29
B10	W/R	F3	GND	L14	D6	P5	D26
B11	GND	F12	CLK2	M1	A26	P6	GND
B12	N.C.	F13	N.C.	M2	A29	P7	D24
B13	BE2	F14	GND	M3	VCC	P8	VCC
B14	GND	G1	A16	M4	GND	P9	D22
C1	A8	G2	VCC	M5	D31	P10	D19
C2	A7	G3	VCC	M6	D28	P11	D18
C3	A6	G12	VCC	M7	VCC	P12	D14
C4	A2	G13	READY	M8	GND	P13	D13
C5	VCC	G14	VCC	M9	D20	P14	GND

Signal Descriptions

The following material describes the 80386 signal lines referred to in the previous two tables. The signals have been grouped according to function for easier identification.

Bus Signals

Data Bus (D0 – D31) — These 32 signal lines are the three-state bidirectional data bus. This bus serves as the general purpose data path between the 80386 and other system components. Data bus inputs and outputs maintain a set (1) condition when in the high logic state.

Address Bus (A2 – A31, $\overline{BE0}$ – $\overline{BE3}$) — The 30 three-state address lines, along with the four byte enable lines, are the address bus for the 80386. This system is slightly different from the more familiar address organization. The byte enable lines are used to directly indicate which bytes of the 32-bit data bus are involved in the current transfer. This scheme can best be seen in the following list:

- If BE0 is asserted, then D0 – D7 are transferred
- If BE1 is asserted, then D8 – D15 are transferred
- If BE2 is asserted, then D16 – D23 are transferred
- If BE3 is asserted, then D24 – D31 are transferred

The byte enable lines can also be combined to generate the more familiar A0 and A1 signals in this computer. Table 12-16 illustrates this signal generation scheme.

Table 12-16. A0, A1 Address Line Generation

A1	A0	BE3	BE2	BE1	BE0
0	0	x	x	x	0
0	1	x	x	0	1
1	0	x	0	1	1
1	1	0	1	1	1

NOTE: x represents a "don't care" condition

If the current memory write cycle or I/O write cycle transfers an operand that occupies only the upper 16 bits of the data bus, duplicate data is simultaneously presented on the corresponding lower 16 bits of the bus. The data duplication pattern depends on the byte enable lines. Table 12-17 illustrates this process.

Table 12-17. Data Duplication as a Function of the Byte Enable Lines

WRITE DATA DUPLICATION				BYTE ENABLE LINE STATUS			
D0-D7	D8-D15	D16-D23	D24-D31	BE0	BE1	BE2	BE3
A	—	—	—	0	1	1	1
—	B	—	—	1	0	1	1
C	—	C	—	1	1	0	1
—	D	—	D	1	1	1	0
A	B	—	—	0	0	1	1
—	B	C	—	1	0	0	1
C	D	C	D	1	1	0	0
A	B	C	—	0	0	0	1
—	B	C	D	1	0	0	0
A	B	C	D	0	0	0	0

NOTES:

- A = logical write data D0-D7
- B = logical write data D8-D15
- C = logical write data D16-D23
- D = logical write data D24-D31
- 1 = high logic level
- 0 = low logic level
- = undefined condition

Bus Arbitration

Bus Hold Request (HOLD) — When this signal is active, a device other than the 80386 is requesting bus control. As long as another device retains bus control this signal will remain high. If RESET is asserted while HOLD is asserted, RESET has priority and will place the bus in an idle state.

Bus Hold Acknowledge ($\overline{\text{HLDA}}$) — When this signal is active, the 80386 has acknowledged the HOLD request and released control of the bus. When $\overline{\text{HLDA}}$ is asserted, the 80386 places all output or bidirectional signals (D0 – D31, A2 – A31, BE0 – BE3, W/R, D/C, M/IO, LOCK, and ADS) in a high-impedance state.

Bus Cycle Definition

The three primary bus cycle definition signals (W/R, D/C, and M/IO) are valid when the ADS signal is asserted. The relationship of these signals to the bus cycle definition is shown in Table 12-18.

Write/Read ($\overline{\text{W/R}}$) — This signal defines the read or write cycle.

Data/Control ($\overline{\text{D/C}}$) — This signal defines a data or control cycle.

Memory/IO ($\overline{\text{M/IO}}$) — This signal defines memory and I/O cycles.

Locked ($\overline{\text{LOCK}}$) — This signal determines whether or not the current bus cycle is locked.

Table 12-18. Bus Cycle Definition

M/IO	D/C	W/R	BUS CYCLE TYPE
0	0	0	Interrupt acknowledge
0	0	1	Does not occur (see note)
0	1	0	I/O data read
0	1	1	I/O data write
1	0	0	Memory code read
1	0	1	Shutdown: Halt:

Table 12-18 (continued). Bus Cycle Definition

M/I/O	D/C	W/R	BUS CYCLE TYPE
1	1	0	Memory data read
1	1	1	Memory data write

NOTE: This condition will occur during idle bus states when ADS is not asserted.

Bus Control

Address Status ($\overline{\text{ADS}}$) — An active condition on this signal indicates that a valid bus cycle definition and address is present on the 80386 pins.

Transfer Acknowledge ($\overline{\text{READY}}$) — This signal indicates that the current bus cycle is complete and active bytes indicated by BE0 – BE3 and BS16 are accepted or provided. This signal is ignored on the first bus state of all bus cycles. Thereafter the signal is sampled each bus state until asserted.

Next Address Request ($\overline{\text{NA}}$) — When this signal is asserted, the 80386 is ready to accept new data from the address and control lines even if the current cycle is not complete. This signal requests address pipelining.

Bus Size ($\overline{\text{BS16}}$) — When this signal is active the 80386 will only use the low-order half (D0 – D15) of the data bus. If the operand is larger than 16 bits, the 80386 will automatically perform another 16-bit bus cycle. If BE2 or BE3 are asserted during a bus cycle and BS16 is also asserted, then the 80386 will adjust to correctly transfer data to the full 32-bit bus.

The CPU

Coprocessor Interface

Coprocessor Request (PEREQ) — When this signal is asserted, the coprocessor is requesting that the 80386 transfer a data operand to or from memory.

Coprocessor Busy ($\overline{\text{BUSY}}$) — If this signal is active, the coprocessor is still executing an instruction. FNINIT and FNCLEX can execute even if BUSY is active. These instructions are used by the coprocessor for initialization and exception clearing. If BUSY is sampled low at the falling edge of reset, then the 80386 will perform an internal self-test.

Coprocessor Error ($\overline{\text{ERROR}}$) — When active, this signal indicates that the previous coprocessor instruction generated a coprocessor error that is not masked by the coprocessor's control register. The 80386 will generate an exception 7 to access error handling software when it encounters this condition. This signal may also be used to determine if an 80387 is present in the system. If ERROR is low no later than 20 CLK2 periods after the falling edge of RESET and remains low until the 80386 begins its first bus cycle, then an 80387 is presumed to be present.

Interrupt Signals

Maskable Interrupt Request (INTR) — If active, this signal indicates a request for interrupt service which can be masked by the 80386 IF bit. When the 80386 responds to this request, it will perform two interrupt acknowledge bus cycles. At the end of the second cycle the 80386 will place an 8-bit vector on the D0 – D7 data lines to indicate the source of the interrupt.

Non-Maskable Interrupt Request (NMI) — This signal indicates an interrupt service request which cannot be masked by software. The 80386 does not use interrupt acknowledge signals when processing an NMI. Once processing on an NMI has started, no further NMIs will be processed until the next IRET instruction.

Miscellaneous

Clock (CLK2) — This signal provides the basic timing for the 80386. The 80386 divides this signal internally to generate the internal processor clock used for instruction execution. Figure 12-21 illustrates the timing of these signals.

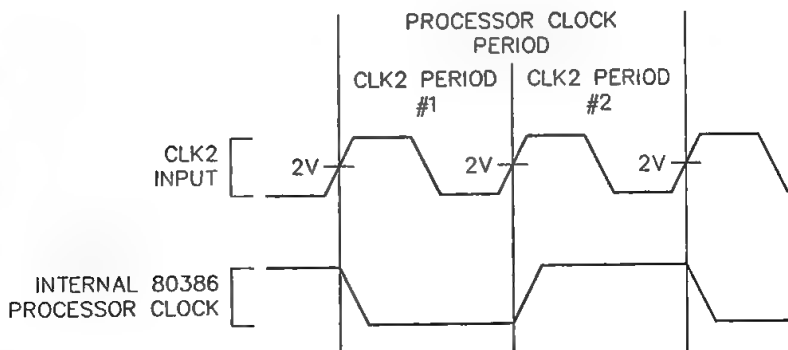


Figure 12-21. Processor Timing Relationships

Reset (RESET) — This signal suspends all processor operations and places the 80386 in a known reset state. When this signal is active, all other input pins are ignored and all other bus pins are driven to an idle state.



Chapter 13

Coprocessors

This chapter provides information on programmer accessible features of the Intel 80287 and 80387 numeric processor extensions (coprocessors). If you require more detailed information concerning these devices, refer to the manufacturer's data sheets. (Refer to Chapter 1 for a list of related publications.)

The 80287 and the 80387 are often referred to as coprocessors. These devices extend and enhance the performance of the system CPU. This is accomplished by allowing the coprocessor to essentially operate in parallel with the CPU. The coprocessors provide additional instructions and perform dedicated mathematical processing tasks.

80287

The 80287 coprocessor (model Z-416-2) is one of two possible coprocessor options that you may install in this computer system. Only one of these two optional devices can be in use in the machine at any time. Depending on which device is in use, some configuration settings may require alteration for correct operation of the coprocessor. Refer to Chapter 4 for details on system configuration.

The 80287 was designed as a complimentary device for the 80286 series of microprocessors. The 80287 operates as an extension of the system CPU for 80286-based and 80386-based systems, providing over 50 additional instructions to the existing CPU instruction set (Refer to Appendix B). Some of the more noteworthy features of this device include:

- 80-bit internal architecture
- Support for the IEEE 754 Floating Point Standard
- Support for multiple data formats (32-, 64-, and 80-bit floating point; 32-, and 64-bit integers; and 18 digit BCD data)
- 8087 object code compatibility.

Coprocessors

This device greatly increases processing speed for certain types of numeric calculations, thereby increasing overall system throughput.

Architecture

Figure 13-1 is a block diagram of the 80287 coprocessor. Refer to this illustration while reading the following material.

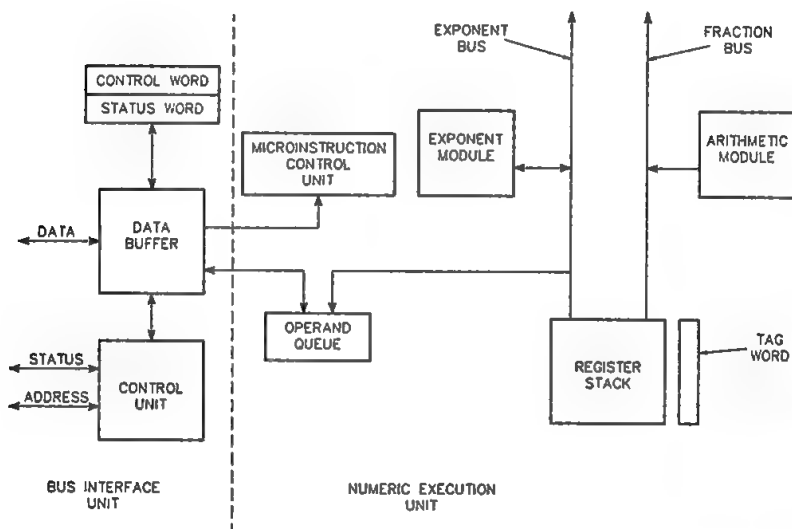


Figure 13-1. 80287 Block Diagram

The internal organization of the 80287 consists of the bus interface unit (BIU) and the numeric execution unit (NEU). These two units perform all processor operations.

Bus Interface Unit

The bus interface unit (BIU) receives and decodes instructions, executes the processor control instructions, and requests operand transfers to and from memory. Each time the CPU encounters an 80287 command, it transmits the instruction to the 80287 for the BIU to decode. If this code is a math instruction, the BIU transfers the information to the NEU. If the code is a control instruction, the BIU executes it independently of the NEU. The BIU also generates the BUSY and ERROR signals used by the CPU.

The status word reports the overall state of the 80287 in a 16-bit word. Table 13-1 describes the bits for this word. Figure 13-2 is a representation of the status word format within the 80287.

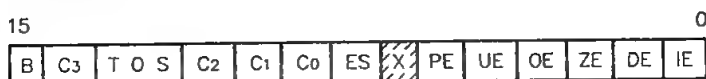


Figure 13-2. 80287 Status Word Format

Table 13-1. 80287 Status Word Bit Definitions

BIT	NAME	DESCRIPTION
0	IE	Invalid operation — stack overflow or underflow, indeterminate form, or use of a non-number as an operand.
1	DE	Denormalized operand — an operand with the smallest exponent, but a non-zero significand.
2	ZE	Zero divide — the dividend is a non-infinite, non-zero number while the divisor is zero.
3	OE	Overflow — the magnitude of the operation result is too large for the specified format.
4	UE	Underflow — the operation result is non-zero but too small to fit the specified format.
5	PE	Precision — is not possible to correctly represent the result in the specified format. The result is rounded.
6	—	Reserved — do not use.

Coprorocessors

Table 13-1 (continued). 80287 Status Word Bit Definitions

BIT	NAME	DESCRIPTION
7	ES	Error summary status — this bit is set if any unmasked exception bit is set.
8–10, 14	C0–C2 C3	Condition codes — report the condition of the operation. Refer to Table 13-2 for the interpretation of these codes.
11–13	TOS	Top of stack — these three bits define the current top of stack according to the following arrangement: 000 = Register 0 current TOS 001 = Register 1 current TOS 010 = Register 2 current TOS 011 = Register 3 current TOS 100 = Register 4 current TOS 101 = Register 5 current TOS 110 = Register 6 current TOS 111 = Register 7 current TOS
15	B	NEU busy — this bit is set whenever the execution unit is busy. This causes the BIU to generate the BUSY signal.

NOTE: If bits 0–5 are set (1), then the condition is true.

Table 13-2. 80287 Condition Codes

INSTRUCTION TYPE	C3	C2	C1	C0	Interpretation
Compare, test	0	0	x	0	TOS > source or 0
	0	0	x	1	TOS < source or 0
	1	0	x	0	TOS = Source or 0
	1	1	x	1	TOS is not comparable
Remainder	Q1	0	Q0	Q2	Complete reduction
	U	1	U	U	Incomplete reduction

Table 13-2 (continued). 80287 Condition Codes

INSTRUCTION TYPE	C3	C2	C1	C0	Interpretation
Examine	0	0	0	0	Valid, positive normalized
	0	0	0	1	Invalid, positive, exponent = 0
	0	0	1	0	Valid, negative, unnormalized
	0	0	1	1	Invalid, negative, exponent = 0
	0	1	0	0	Valid, positive, normalized
	0	1	0	1	Infinity, positive
	0	1	1	0	Valid, negative, normalized
	0	1	1	1	Infinity, negative
	1	0	0	0	Zero, positive
	1	0	0	1	Not used
	1	0	1	0	Zero, negative
	1	0	1	1	Not used
	1	1	0	0	Invalid, positive, exponent = 0
	1	1	0	1	Not used
	1	1	1	0	Invalid, negative, exponent = 0
	1	1	1	1	Not used

NOTES:

x = unaffected value

U = undefined value

TOS = top of stack

QX = quotient bit X

The control word determines which of several different processing options the coprocessor will use. The 16-bit control word is normally loaded from memory to set specific coprocessor options. The five low-order bits (0 – 5) of the word contains individual masks for each of the six exceptions that the 80287 recognizes. If an exception is encountered during processing and the mask is in place, processing will continue as normal. If the exception is not masked, then the 80287 will generate an error condition. The eight high-order bits of the word contain operating parameter controls such as precision, rounding, and infinity. Table 13-3 describes the bit definitions for the control word and Figure 13-3 represents the format of the control word.

Coprorocessors



Figure 13-3. 80287 Control Word Format

Table 13-3. 80287 Control Word Bit Definitions

BIT	NAME	DESCRIPTION
0	IM	Invalid operation exception mask. When the bit is set, the exception is masked.
1	DM	Denormalized operand exception mask. When the bit is set, the exception is masked.
2	ZM	Zero divide exception mask. When this bit is set, the exception is masked.
3	OM	Overflow exception mask. When the bit is set, the exception is masked.
4	UM	Underflow exception mask. When the bit is set, the exception is masked.
5	PM	Precision exception mask. When the bit is set, the exception is masked.
6, 7	—	Reserved. Do not use.
8, 9	PC	Precision control. These bits determine the precision of processor calculations according to the following settings: <div style="margin-left: 40px;"> 00 = 24 bits (short real) 01 = reserved 10 = 53 bits (long real) 11 = 64 bits (temp real) </div>
10, 11	RC	Rounding control. These bits determine the mode used for rounding operations during calculations according to the following settings: <div style="margin-left: 40px;"> 00 = round to nearest or even 01 = round down 10 = round up 11 = Chop (truncate toward zero) </div>

Table 13-3 (continued). 80287 Control Word Bit Definitions

BIT	NAME	DESCRIPTION
12	IC	Infinity control. The setting of this bit determines how the processor controls the closure of the number space at infinity. When the bit is set, affine closure is used (\pm infinity). When the bit is clear, projective closure is used.
13 - 15	—	Reserved. Do not use.

Numeric Execution Unit

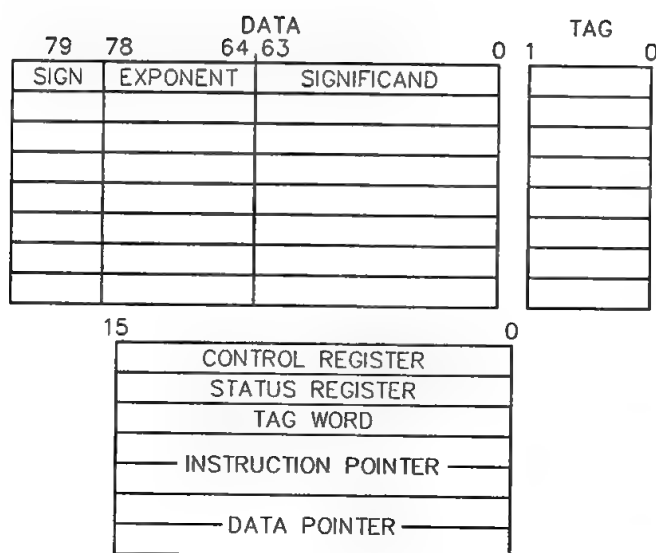
The numeric execution unit (NEU) is the second major portion of the 80287. This unit is responsible for executing all instructions dealing with the register stack, including logical, arithmetic, constant, transcendental, and data transfer instructions. The internal data path in the NEU is 84 bits wide. This path consists of 68 significand bits, 15 exponent bits, and 1 sign bit.

The NEU contains the register stack, all numeric execution units for the processor, the operand queue, and the microcode control unit. Of all the components within the NEU, the most important is the register stack.

Register Resources

The 80287 employs a stack consisting of eight data registers. Each register is 80 bits wide and contains three fields. The lower 64 bits of each register comprise the significand data, the next 14 bits are the exponent data, and the highest bit is the sign bit. Figure 13-4 illustrates the registers within the 80287.

Coprorocessors



Reprinted by permission of Intel Corporation, Copyright 1987

Figure 13-4. 80287 Registers

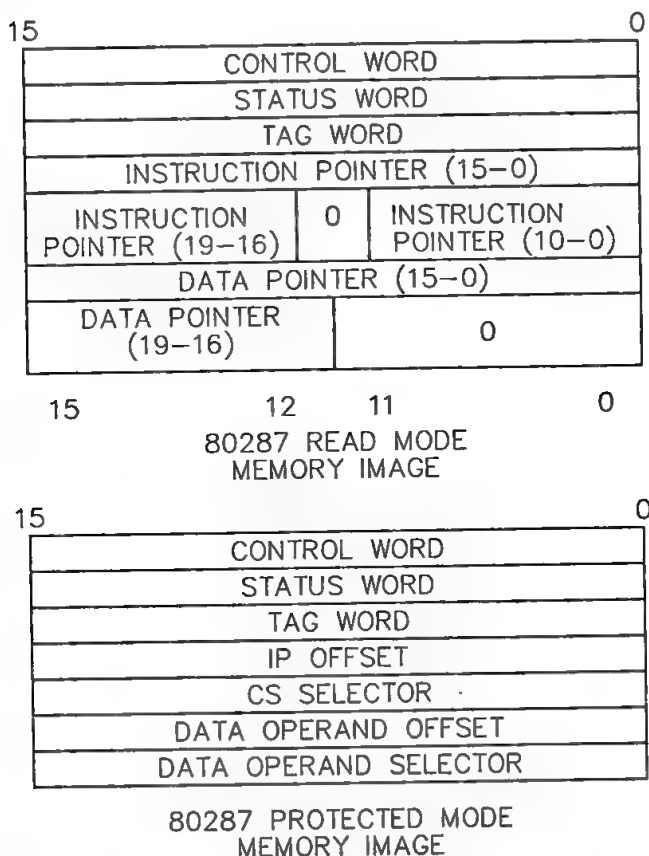
Each register has a two-bit tag associated with it. All eight tags are contained in the tag word register. The two lowest-order bits are the tag for register 0 and the two highest-order bits are the tag for register 7. There are four possible tag values, as follows:

- 00 — (valid tag)
- 01 — (zero tag)
- 10 — (invalid tag or infinity)
- 11 — (empty tag)

These tag values help to determine the contents of the register.

Coprorocessors

The register set also contains an instruction pointer and a data pointer register. These registers are useful for error-handling routines. Since the 80287 supports both real and protected mode operation, these registers can be interpreted in either of two ways. Figure 13-5 illustrates the differences between these two modes.



Reprinted by permission of Intel Corporation, Copyright 1987

Figure 13-5. Register Mode Differences

Coprocessors

Operation

The 80287 operates in parallel with the system processor. The BIU decodes each instruction that the system CPU receives. If it receives an 80287 instruction, it determines if the instruction is a control instruction or a math instruction. The BIU executes control instructions and transfers math instructions to the NEU.

When the NEU executes an instruction, it causes the BIU to activate the processor's BUSY line. Since the 80287 only executes one instruction at a time, the system CPU checks the status of the BUSY line before sending commands to the 80287. If the 80287 encounters an exception during program execution, it will cause the BIU to activate the ERROR line and it will set the ES bit in the status word.

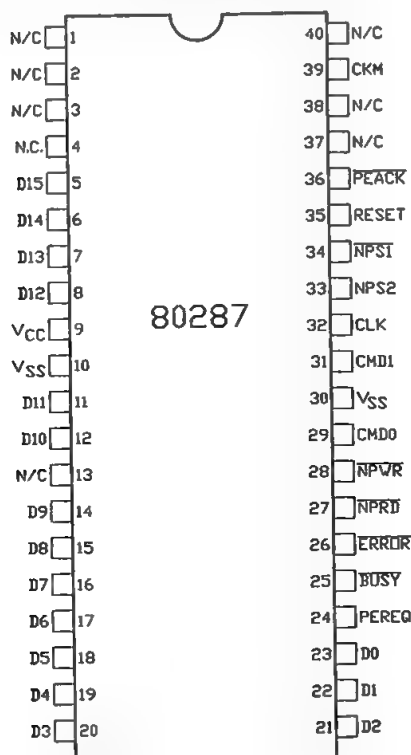
The 80287 supports two separate operating modes similar to those of the 80286 microprocessor. When the 80287 is reset, it automatically sets up to run in real address mode. The 80287 can be placed in the protected virtual address mode by executing the FSETPM instruction. Note that once the 80287 is placed in protected mode, it cannot return to real mode without being reset. Also, when the 80287 is in protected, mode all references to memory for data status obey 80286 protection rules. The 80287 is compatible with the 8087 in both modes of operation.

Programming

Programming the 80287 is a very simple process. The device is controlled and accessed through the instruction set (refer to Appendix B). Even the status word and the control word are handled through the processor instruction commands. This allows the programmer to include specific 80287 routines with any number of high-level languages or with other assembly language routines.

Device Pinout

Figure 13-6 illustrates the pinout of the 80287 coprocessor. Refer to Table 13-4 for descriptions of these signals.



Reprinted by permission of Intel Corporation, Copyright 1987

Figure 13-6. 80287 Signal Pinout

Table 13-4. 80287 Pin Descriptions

PIN	NAME	DESCRIPTION
1-4	—	Not connected.
5-8	D15-D12	Data bus. Bidirectional data bus signals.
9	V _{CC}	+5VDC power supply.
10, 30	V _{SS}	System ground.

Coprorocessors

Table 13-4 (continued). 80287 Pin Descriptions

PIN	NAME	DESCRIPTION
11, 12	D11, D10	Data bus. Bidirectional data bus signals.
13	—	Not connected.
14–23	D9–D0	Data bus. Bidirectional data bus signals.
24	PEREQ	Processor extension data channel operand transfer request. When this line is high, the 80287 is ready to transfer data. This signal is disabled when $\overline{\text{PEACK}}$ goes active.
25	$\overline{\text{BUSY}}$	Busy status. This line is active whenever the 80287 is executing a command.
26	$\overline{\text{ERROR}}$	Error status. This signal is active when an unmasked error condition exists in the 80287.
27	$\overline{\text{NPRD}}$	Numeric processor read. When this signal is active, data may be read from the 80287.
28	$\overline{\text{NPWR}}$	Numeric processor write. When this signal is active, data may be written to the 80287.
29	CMD0	Command line 0. The system CPU uses this line in conjunction with CMD1, $\overline{\text{NSP1}}$, and NSP2 to operate the 80287.
31	CMD1	Command line 1. The system CPU uses this line in conjunction with CMD0, $\overline{\text{NSP1}}$, and NSP2 to operate the 80287.
32	CLK	Clock input. The basic timing signal for the 80287 is provided on this input.
33	NSP2	Numeric processor select. The system CPU uses this line in conjunction with CMD0, $\overline{\text{NSP1}}$, and CMD1 to operate the 80287.
34	$\overline{\text{NSP1}}$	Numeric processor select. The system CPU uses this line in conjunction with CMD0, $\overline{\text{NSP1}}$, and CMD1 to operate the 80287.
35	RESET	System reset. This signal causes the 80287 to terminate all present activity and return to the real address mode.
36	$\overline{\text{PEACK}}$	Processor extension data channel operand transfer acknowledge. This signal acknowledges the PEREQ signal.
37, 38	—	Not connected.
39	CKM	Clock mode signal. A high on this line causes the CLK signal to be used directly for processor timing. A low on this signal causes the CLK signal to be divided by three before use.
40	—	Not connected.

80387

The 80387 coprocessor (model Z-516) is the second of two coprocessor options available for this computer system. As with the 80287, installation of this coprocessor may require alteration of the system configuration. Refer to Chapter 4 for details on system configuration.

The 80387 serves as a complimentary device for 80386-based computer systems. Like the 80287, the 80387 operates as an extension of the 80386 CPU, providing it with additional instructions and features (refer to Appendix C). Some of the features of this device include:

- 80-bit internal architecture
- Object code compatibility with 8087 and 80287
- Support for multiple data formats (32-, 64, and 80-bit floating point; 32-, and 64-bit integers; and 18 digit BCD data)
- Support for the ANSI/IEEE 754-1985 Binary Floating-Point Standard

Architecture

Figure 13-7 is a block diagram of the 80387 coprocessor. Refer to this illustration while reading the following material.

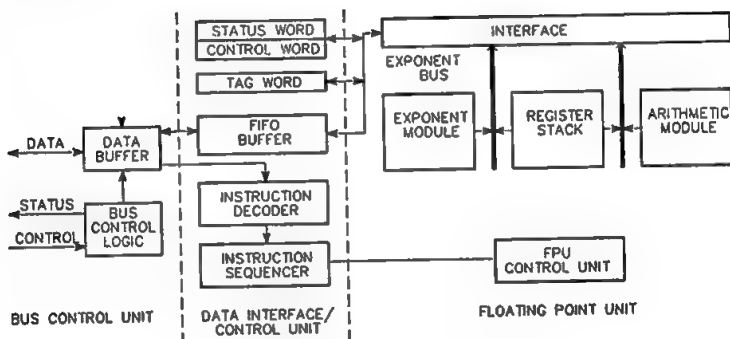


Figure 13-7. 80387 Block Diagram

Coprocessors

The internal organization of the 80387 is similar to the 80287. Design changes within the 80387 allow the 80386 CPU to perform some of the operations that the 80287 previously had to complete. This results in increased performance in the 80387, providing faster processing.

Internally, the 80387 consists of three major circuit groups. The three major groups are the bus control logic unit, the data interface/control unit and the floating point unit. These units, combined with some functions within the 80386, perform the dedicated math processing functions. In order to enhance performance, the three sections operate in parallel.

Bus Control Unit

The bus control unit interfaces the 80387 with the 32-bit data bus and controls the operation of that interface. As far as the CPU is concerned, the 80387 appears as a special peripheral device. Whenever the CPU encounters a coprocessor command, it automatically initiates I/O with the 80387 using reserved I/O addresses.

The bus control unit communicates only with the CPU. The CPU, in turn, handles all memory accesses required by the 80387, transferring data to and from the coprocessor.

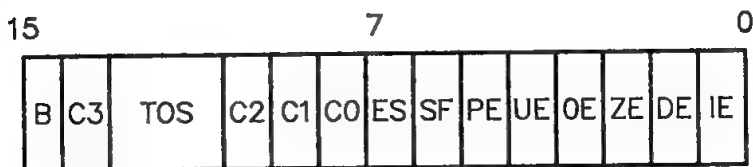
Data Interface/Control Unit

The data interface/control unit controls the flow of data to and from the floating point unit and control registers. It also decodes instructions, sequences internal microinstructions, and performs some internal administrative functions.

The data interface/control unit reads data from the bus control logic area. Under the control of the bus control logic, this data is sent to the instruction decoder or an internal FIFO buffer. The instruction decoder controls the flow of data in the FIFO buffer as well as the microinstructions that execute each instruction. Some 80387 instructions can execute independently of the floating point unit and the

microinstruction sequencer. The instructions in this group are FINIT, FCLEX, FSTSW, FSTSW AX and FSTCW (refer to Appendix C for information on the 80387 instruction set). This unit also generates the BUSY, PEREQ, and ERROR signals to synchronize operations with the CPU.

The data interface/control unit uses the status word register to report the status of the coprocessor after each operation. The status word is a 16-bit word, as illustrated in Figure 13-8. Table 13-5 describes the bit definitions for this word. This status word differs from that used by the 80287 only in the definition of bit 6.



Reprinted by permission of Intel Corporation, Copyright 1987

Figure 13-8. 80387 Status Word Format

Table 13-5. 80387 Status Word Bit Definitions

BITS	NAME	DESCRIPTION
0	IE	Invalid operation — stack overflow or underflow, indeterminate form, or use of a non-number as an operand.
1	DE	Denormalized operand — an operand with the smallest exponent, but a non-zero significant.
2	ZE	Zero divide — the dividend is a non-infinite, non-zero number while the divisor is zero.
3	OE	Overflow — the magnitude of the operation result is too large for the specified format.
4	UE	Underflow — the operation result is non-zero but too small to fit the specified format.

Coprorocessors

Table 13-5 (continued). 80387 Status Word Bit Definitions

BIT	NAME	DESCRIPTION
5	PE	Precision — is not possible to correctly represent the result in the specified format. The result is rounded.
6	SF	Stack flag — This bit helps to distinguish between invalid operations involving the stack. When this bit is set, an invalid operation has occurred. If bit 9 is set, the operation was a stack overflow. If bit 9 is clear, the operation was a stack underflow.
7	ES	Error summary status — this bit is set if any unmasked exception bit is set. If this bit is set, the $\overline{\text{ERROR}}$ signal is active.
8 – 10, 14	C0 – C2 C3	Condition codes — report the condition of the operation. Refer to Table 13-6 for the interpretation of these codes.
11 – 13	TOS	<p>Top of stack — these three bits define the current top of stack according to the following arrangement:</p> <p>000 = Register 0 current TOS 001 = Register 1 current TOS 010 = Register 2 current TOS 011 = Register 3 current TOS 100 = Register 4 current TOS 101 = Register 5 current TOS 110 = Register 6 current TOS 111 = Register 7 current TOS</p>
15	B	Busy bit — this bit is provided only for compatibility with the 8087. The condition of this bit reflects the contents of the ES bit.

NOTE: If bits 0 – 5 are set (1), then the condition is true.

Table 13-6. 80387 Condition Codes

INSTRUCTION TYPE	C3	C2	C1	C0	Interpretation
Compare	0	0	x	0	TOS > operand
	0	0	x	1	TOS < operand
	1	0	x	0	TOS = operand
	1	1	x	1	Unordered
Remainder	Q1	0	Q0	Q2	Complete reduction (See Table 13-7)
	U	1	U	U	Incomplete reduction
Operand Class Definitions	0	0	0	0	+ Unsupported
	0	0	0	1	+ NAN
	0	0	1	0	- Unsupported
	0	0	1	1	- NAN
	0	1	0	0	+ Normal
	0	1	0	1	+ Infinity
	0	1	1	0	- Normal
	0	1	1	1	- Infinity
	1	0	0	0	+ 0
	1	0	0	1	+ Empty
	1	0	1	0	- 0
	1	0	1	1	- Empty
	1	1	0	0	+ Denormal
	1	1	1	0	- Denormal

NOTES:

x = unaffected value

U = undefined value

TOS = top of stack

Qx = quotient bit x

NAN = not a number

The NAN listed in Table 13-6 is a special condition that occurs in the 80387 coprocessor. Any value that contains a string of ones in the exponent, except infinity, is considered a NAN. This situation tends to propagate through arithmetic operations. The advantage to NAN values is that they can be handled as special situations by the software. This allows the programmer to test for special conditions or results with the coprocessor.

Coprocessors

Complete reductions, as listed in Table 13-6, relate to a specific MOD value. Table 13-7 provides the relation between the condition codes, quotient bits, and the MOD value.

Table 13-7. 80387 Quotient Results

CONDITION CODE QUOTIENT BIT	C3	C2 Q1	C1 Q0	C0 Q2	VALUE Q MOD 8
	0	0	0	0	0
	0	0	1	0	1
	0	1	0	0	2
	0	1	1	0	3
	0	0	0	1	4
	0	0	1	1	5
	0	1	0	1	6
	0	1	1	1	7

The 80387 also employs a control word, as in the 80287. However, in the 80387 bit 12 no longer defines infinity control. Instead, it is defined as a reserved bit. (Refer to Table 13-3 for details on the other bits of the control word.) In the 80387, only affine closure is supported in infinity arithmetic. Bit 12 initializes to a zero after a RESET or an FINIT instruction and can only be changed by loading the control word. Since this bit is defined as reserved, software should ignore this bit.

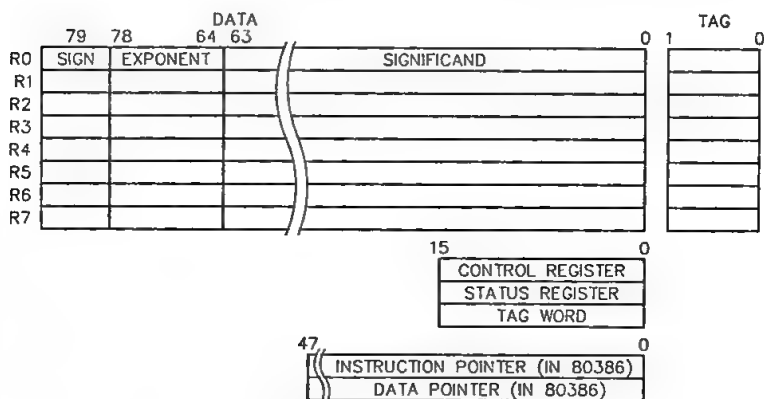
Floating Point Unit

The floating point unit executes all numerics instructions. Numerics instructions are those instructions that involve the processor stack. They include arithmetic, logical, transcendental, constant, and data transfer instructions. As in the 80287, the internal data path in the floating point unit is 84 bits wide. The data path is divided into three sections: 68 bits are the significant bits, 15 bits are exponential, and one bit in the path is the sign bit.

Register Resources

The 80387 uses a register set that is very similar to the register set in the 80287. The primary difference between the two are the instruction pointer and the data pointer registers.

In the 80287, both the instruction pointer and the data pointer registers are part of the register set. The 80387 has the same capabilities, but these registers are no longer within the 80387 itself. Instead, the two registers are located within the 80386 CPU. Because of this, the size of these registers has increased from 16 bits to 48 bits. Figure 13-9 illustrates the register set used by the 80387.



Reprinted by permission of Intel Corporation, Copyright 1987

Figure 13-9. 80387 Registers

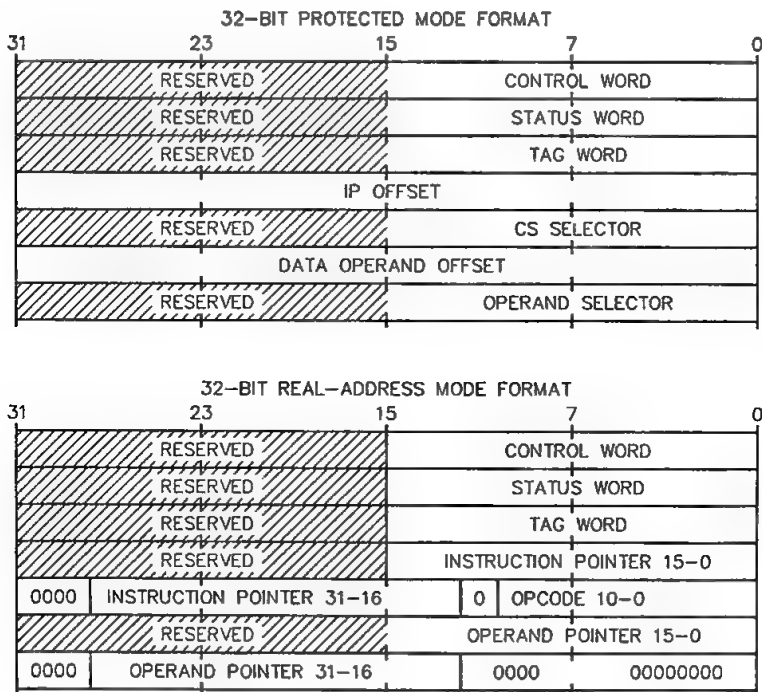
As in the 80287, each register has a two-bit tag associated with it. All eight tags are contained in the tag word register. The two lowest-order bits are the tag for register 0 and the two highest-order bits are the tag for register 7. There are four possible tag values, as follows:

- 00 — (valid tag)
- 01 — (zero tag)
- 10 — (invalid tag or infinity)
- 11 — (empty tag)

These tag values help to determine the contents of the register.

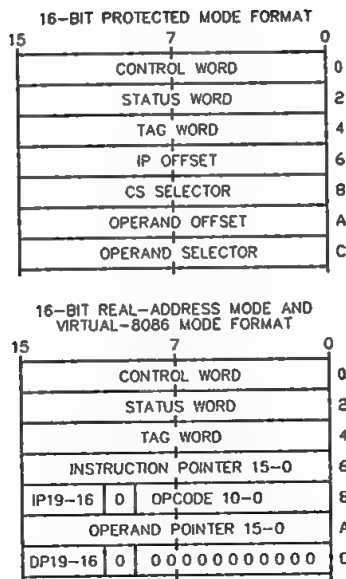
Coprocessors

Like the 80287, the 80387 supports different modes of operation. The 80387 supports both real and protected mode operations in 16- and 32-bit formats. Because of this, the interpretation of the various registers may change based on the current operating mode. Figure 13-10 illustrates the differences in 32-bit format while Figure 13-11 illustrates the differences in 16-bit format.



Reprinted by permission of Intel Corporation, Copyright 1987

Figure 13-10. 32-bit Format Register Differences



Reprinted by permission of Intel Corporation, Copyright 1987

Figure 13-11. 16-Bit Format Register Differences

Operation

In operation, the 80387 is very similar to the 80287. As with the 80287, the 80387 operates in parallel with the 80386. Essentially, the 80386 looks at the coprocessor as an extension of itself. In both real and virtual 8086 mode, the 80387 is upward compatible with the 8087 and the 80287. In the protected mode the 80387 is upward compatible with the 80287.

Coprocessors

Whenever the 80386 CPU encounters a coprocessor command, it communicates with the bus control logic of the 80387. The 80387 then latches the instruction into the data buffer so that the data interface/control unit can access it. The data interface/control unit passes the data to the FIFO buffer or to the instruction decoder, depending on the instructions received from the bus control unit. When the microinstruction sequencer starts to execute the instruction, the 80387 BUSY line goes active. If the 80387 encounters a processing exception during execution, the ERROR line will go active.

All computations, with the exception of those handled by the data control/interface unit, are performed by the floating point unit. The results of these calculations are routed back to the FIFO buffer so that they can be placed back on the system data bus.

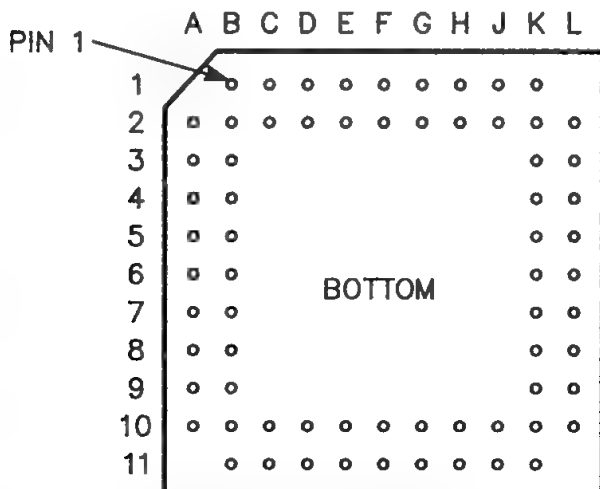
Programming

The 80387 is programmed in the same manner as the 80287. All device and control functions are accessible through the instruction set (refer to Appendix C). Since the 80386 controls the 80387, all communications between the two devices will be transparent to applications software. The device can be utilized by high-level and assembly language routines.

All operands are stored in memory with the least-significant digit at the lowest memory address. To increase system performance, all operands should start at doubleword boundaries. A doubleword boundary is a memory address that is evenly divisible by four. This allows the processor to utilize the full 32-bit data path while moving data.

Device Pinout

Figure 13-12 illustrates the pinout of the 80387 coprocessor. Table 13-8 is a cross-reference of the device pinout by signal name, and Table 13-9 provides the same cross-reference by pin number.



Reprinted by permission of Intel Corporation, Copyright 1987

Figure 13-12. 80387 Device Pinout

Table 13-8. 80387 Pinout by Function

FUNCTION PIN NAME		FUNCTION PIN NAME		FUNCTION PIN NAME	
D0	H2	D23	C10	387CLK2	K11
D1	H1	D24	D10	Vcc	A6
D2	G2	D25	D11	Vcc	A9
D3	G1	D26	E10	Vcc	B4
D4	D2	D27	E11	Vcc	E1
D5	D1	D28	G10	Vcc	F1
D6	C2	D29	G11	Vcc	F10
D7	C1	D30	H10	Vcc	J2
D8	B1	D31	H11	Vcc	K5
D9	A2	<u>ADS</u>	K7	Vcc	L7
D10	B3	<u>BUSY</u>	K2	GND	B2
D11	A3	CKM	J11	GND	B7

Coprocessors

Table 13-8 (continued). 80387 Pinout by Function

FUNCTION PIN NAME		FUNCTION PIN NAME		FUNCTION PIN NAME	
D12	A4	$\overline{\text{CMD0}}$	L8	GND	C11
D13	B5	$\overline{\text{ERROR}}$	L2	GND	E2
D14	A5	$\overline{\text{NPS1}}$	L6	GND	F2
D15	B6	$\overline{\text{NPS2}}$	K6	GND	F11
D16	A7	$\overline{\text{PEREQ}}$	K1	GND	J1
D17	B8	$\overline{\text{READY}}$	K8	GND	J10
D18	A8	$\overline{\text{READY0}}$	L3	GND	L5
D19	B9	$\overline{\text{RESETIN}}$	L10	N.C.	K9
D20	B10	STEN	L4	Tie High	K3
D21	A10	$\overline{\text{W/R}}$	K4	Tie High	L9
D22	B11	386CLK2	K1		

Table 13-9. 80387 Pinout by Pin Name

PIN NAME FUNCTION		PIN NAME FUNCTION		PIN NAME FUNCTION	
A2	D9	C11	GND	J10	GND
A3	D11	D1	D5	J11	CKM
A4	D12	D2	D4	K1	$\overline{\text{PEREQ}}$
A5	D14	D10	D24	K2	BUSY
A6	Vcc	D11	D25	K3	Tie High
A7	D16	E1	Vcc	K4	$\overline{\text{W/R}}$
A8	D18	E2	GND	K5	Vcc
A9	Vcc	E10	D26	K6	$\overline{\text{NPS2}}$
A10	D21	E11	D27	K7	$\overline{\text{ADS}}$
B1	D8	F1	Vcc	K8	$\overline{\text{READY}}$
B2	GND	F2	GND	K9	N.C.
B3	D10	F10	Vcc	K10	386CLK2
B4	Vcc	F11	GND	K11	387CLK2
B5	D13	G1	D3	L2	$\overline{\text{ERROR}}$
B6	D15	G2	D2	L3	$\overline{\text{READY0}}$
B7	GND	G10	D28	L4	STEN
B8	D17	G11	D29	L5	GND
B9	D19	H1	D1	L6	$\overline{\text{NPS1}}$
B10	D20	H2	D0	L7	Vcc
B11	D22	H10	D30	L8	$\overline{\text{CMD0}}$
C1	D7	H11	D31	L9	Tie High
C2	D6	J1	GND	L10	$\overline{\text{RESETIN}}$
C10	D23	J2	Vcc		

80387 Pin Descriptions

This section is organized according to the signal groups used in the 80387. These groups are control signals, handshake signals, bus interface signals, chip select signals, and power connections.

Control Signals

The control signals are responsible for all execution control in the 80387.

386CLK2 — 80386 clock 2. The operation of this signal depends on the setting of J201. If the 80387 is set for non-synchronous operation, this signal provides the clock signal for the bus control logic. If the 80387 is operating in synchronous mode, this signal also clocks the floating point unit and the data interface/control unit. This signal is divided by two internally to form the internal clock.

387CLK2 — 80387 clock 2. This signal also depends on the setting of J201. When the 80387 is in asynchronous mode, this signal provides the clock signal for the data interface/control unit and the floating point unit. If the 80387 is operating in synchronous mode, this pin is ignored.

CKM — Clocking mode. This pin determines which clocking mode the 80387 will use. The two available modes are synchronous or asynchronous. This line is controlled by the setting of J201.

RESETIN — System reset. When the signal level on this pin changes from low to high, the 80387 will terminate all present operations and revert to a dormant state.

Handshake Signals

The handshake signals are responsible for reporting the status of the 80387.

Coprocessors

PEREQ — Processor extension request. Whenever the 80387 is ready to transfer data to or from the FIFO buffer, this line will go active. This line is normally connected directly to the 80386 CPU.

BUSY — Busy status. Whenever the 80387 is executing an instruction, this line will be active.

ERROR — Error status. If the 80387 encounters an unmasked exception during execution, this line will go active. This line reflects the state of the ES bit in the status register. Immediately following a system reset, the ES bit is read to determine if an 80387 is present in the system.

Bus Interface Signals

The bus interface signals control reading and writing information to and from the system bus.

D0 – D31 — Bidirectional data bus. All data going to and from the 80387 uses this bus. These lines are normally connected directly to the data bus for the 80386.

W/R — Write/read cycle. This line lets the 80387 know what type of bus cycle is currently in progress. If $\overline{\text{STEN}}$, $\overline{\text{NSP1}}$, or $\overline{\text{NSP2}}$ are inactive, this signal will be ignored by the 80387.

ADS — Address strobe. This signal is used with the $\overline{\text{READY}}$ signal by the bus control logic to determine when the 80387 can sample the $\overline{\text{W/R}}$ and chip select signals.

READY — Bus ready. Whenever the 80386 is about to terminate a bus cycle, this signal will become active. The internal bus control logic uses this signal follow activity on the bus.

READY0 — Ready output. When this signal is active, it indicates that the current 80387 bus cycle may be terminated if no additional wait states are required.

Chip Select Signals

The chip select signals allow the system to select the 80387 for operations.

STEN — Status enable. This line is essentially the chip select line for the 80387. This signal requires that $\overline{\text{NSP1}}$ and $\overline{\text{NSP2}}$ be active in order to correctly select the 80387.

$\overline{\text{NSP1}}$ — 80387 select #1. When this line is active along with STEN and $\overline{\text{NSP2}}$, it indicates that the current bus cycle is communicating with the 80387. This line is selected when the 80386 performs I/O cycles.

$\overline{\text{NSP2}}$ — 80387 select #2. When this line is active along with STEN and $\overline{\text{NSP1}}$, it indicates that the current bus cycle is communicating with the 80387.

$\overline{\text{CMD0}}$ — Command. This signal indicates whether the 80387 has received an opcode or data. If $\overline{\text{CMD0}}$ is active, the command was an opcode for the 80387. If $\overline{\text{CMD0}}$ is inactive, the information is data for the 80387.

Power Signals

The final group of signals are those that provide operating power for the 80387.

Vcc — +5 VDC supply. This is the operating DC voltage for the 80387.

GND — Ground. All signals with this designation connect to system ground.



Chapter 14

Support Circuits

This chapter describes the major user-programmable support circuits in this computer. Support circuits remove some of the workload from the system CPU by maintaining critical system operations without requiring direct CPU control. Examples of support circuits are DMA controllers, interrupt controllers, and system timing circuits.

CPU Support Circuitry

Several integrated circuits make up most of the CPU support circuitry. Support circuits include two 8259 interrupt controllers, two 8237 DMA controllers, the MC146818A real-time clock, the 8254 programmable interval timer, and a dedicated processor known as the system control processor, or SCP. A number of data latches and buffers are also contained in this circuitry.

With the exception of the real-time clock, all of these components are located on the system I/O card. The real-time clock is installed on the backplane board. The system I/O card also contains the serial and parallel communications circuits. These circuits are discussed in other chapters in this manual. The I/O card occupies slot number 4 in the backplane board.

The 8259 Interrupt Controller

The interrupt controllers monitor various devices and the system bus and notify the CPU if any of them require immediate attention. These devices include the timer, keyboard, real-time clock, the co-processor, and the input/output ports.

This computer uses two types of interrupts: maskable and non-maskable interrupts. The interrupt controller handles the maskable interrupts while non-maskable interrupts go directly to the CPU. The non-maskable interrupt is reserved for serious error conditions and those circuits that have critically tight timing requirements.

Support Circuits

The design of the 8259 is compatible with the interrupt-driven environment in this computer. Each 8259 can manage up to eight separate request lines. Since this computer uses the interrupt controllers in cascade mode, the slave controller uses one line as an input to the master controller. This provides a total of 15 separate interrupt lines that the system can use.

You may select from several priority modes that configure the 8259 to match the hardware requirements of the system. This configuration may be changed while the system is running and allows the user to re-configure the system as various requirements for interrupts arise during operation of the system.

The user can mask individual interrupt lines without affecting those with either higher or lower priority. Table 14-1 shows the interrupt level assignments available in this computer.

Table 14-1. Interrupt Line Assignments

INTERRUPT	ASSIGNMENT
IRQ0	Timer channel 0
IRQ1	System control processor (SCP) port 2, bit 4
IRQ2	Slave controller input
IRQ3	PC bus or COM2
IRQ4	PC bus or COM1
IRQ5	PC bus or LPT2
IRQ6	PC bus
IRQ7	PC bus or LPT1
IRQ8	Real-time clock
IRQ9	IRQ2 on PC bus
IRQ10	AT bus
IRQ11	AT bus
IRQ12	AT bus
IRQ13	Coprocessor
IRQ14	AT bus
IRQ15	AT bus

The system timer interrupt, which has the highest priority of the maskable interrupts, is a clock signal generated by the 8254. The computer uses it for disk drive timing and to generate certain clock signals used by the operating system. The keyboard interrupt, controlled by the SCP, has the next highest priority. The keyboard interrupt occurs whenever the user presses a key on the keyboard.

Interrupt Controller Architecture

Refer to the block diagram in Figure 14-1 while reading the following material. There are eight major circuit groups in the 8259 that the computer uses. Lines and arrows illustrate the interrelationship between these blocks.

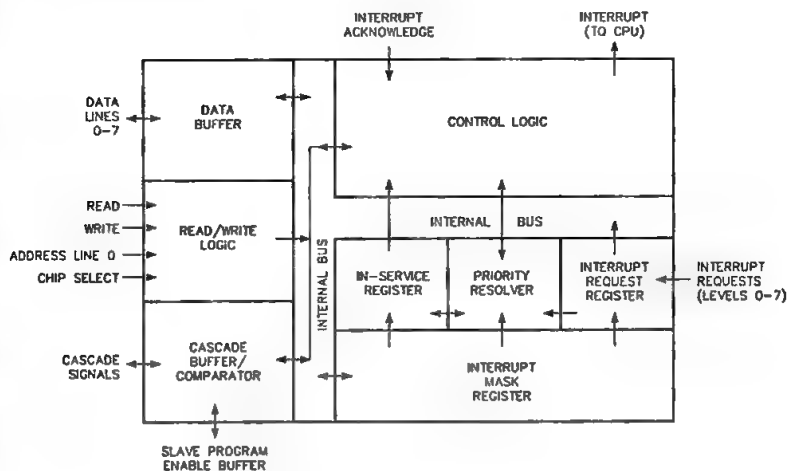


Figure 14-1. 8259 Interrupt Controller Block Diagram

The data buffer is an 8-line, bidirectional, three-state buffer that interfaces the address/data bus to the CPU. Instructions, status information, and interrupt vector data are transferred through this buffer.

Support Circuits

The control logic is responsible for sending the interrupt signal to the CPU and receiving the interrupt acknowledge back from the CPU. It communicates with the priority resolver and receives information from the in-service register. During initialization and programming, the control logic receives instructions from the read/write logic.

The read/write logic is responsible for handling the programming of the chip, both during initialization and during operation. It is also responsible for releasing status information onto the data bus when requested by the CPU. The controlling lines, treated by the CPU as an input/output port, are the read, write, chip select, and address line 0 lines.

The in-service register is one of three internal registers in the IC. This register stores all interrupt requests acknowledged by the CPU that are currently being serviced. The priority resolver looks at this data to see if an incoming interrupt request is already being serviced by the CPU.

The priority resolver determines the priorities of the interrupt requests in the interrupt request register. It first determines if the interrupt request line has been masked by the interrupt mask register. Next, it checks if the interrupt is in the in-service register, which tells the resolver that the interrupt is already being serviced.

The interrupt request register receives all the incoming interrupt request lines and latches them until the CPU can process them. This register stores all interrupts that are requesting service. After the CPU acknowledges an interrupt request, it transfers the request to the in-service register. However, before the interrupt request register will recognize another interrupt request on the same line, that line must go inactive. If the in-service register already holds the request, the interrupt request register will clear (which also clears the in-service register) and will recognize a subsequent interrupt request on the same line. This action prevents an interrupt request that has been serviced by the CPU from reactivating the service routine by going through the system a second time.

The cascade buffer/comparator is responsible for comparing the IDs of all 8259s used in the computer. When the 8259 is used in the master/slave mode, the master sends slave IDs over the cascade lines (CAS0 – CAS2). If the selected slave has an interrupt pending, it will place its subroutine address on the data bus.

The interrupt mask register can disable selected interrupt request lines. It operates in conjunction with the priority resolver to mask out incoming interrupt requests. Masking one priority level will not affect the operation of higher or lower priority interrupts.

Initialization

As part of initializing the computer system, the CPU must initialize certain internal registers and program the interrupt controller by sending it a series of commands.

The system will first initialize the SS (stack segment) register of the CPU. This register points to the start of the stack, which is located near the top of scratchpad memory between addresses F0000H and F3FFFH. The exact address depends on the version of the Monitor program installed in your computer. Note that only the Monitor program uses this stack location. Under MS-DOS operation, the stack segment register may point elsewhere.

Next, the interrupt vector table in low memory is initialized. This table contains the addresses of the service routines used by each interrupt. If the CPU is operating in real mode, each interrupt vector address is four bytes long. The CS (code segment) register uses two bytes for its base address, while the IP (instruction pointer) uses the other two bytes for the offset address. If the processor is operating in protected mode, each vector consists of 8 bytes of information. This is combined with the address in the interrupt descriptor table to obtain the vector address.

Support Circuits

Finally, the CPU will initialize the interrupt controller with a series of initialization commands. These commands set up the controller for normal operation. Tables 14-2 through 14-5 describe the contents of the four command words that may be sent to the controller. Note that either three or four commands may be sent in sequence. The A0 line, when low, identifies the first command word. The A0 line is then set high and the following command words are sent to the controller in sequence.

Table 14-2. Initialization Command Word 1

BIT	DESCRIPTION
0	Informs the controller whether there will be three or four command words in the sequence. If the bit is high, there will be four command words in the sequence. If the bit is low, there will be three command words in the sequence.
1	Tells the controller whether it is the only controller in the system or not. If this bit is set (1), only one controller is in the system and no command word 3 will be issued.
2	Tells the controller whether the vector addresses are four bytes long or eight bytes long. This computer uses four-byte vector addresses, so this bit will be set (1).
3	Tells the controller to recognize interrupt requests on the leading edge of the interrupt line transition or when the level is high. If this bit is set (1), the interrupt request is recognized on the edge of the transition.
4	Always set (1).
5-7	Not used in this computer. If the computer using the 8259 contains an 8-bit processor, such as the 8080 or 8085, then these three bits are programmed with address bits A5 - A7 of the vector table address. If the vector addresses are four bytes long, the interrupt controller will automatically supply address bits A0 - A4, depending upon the interrupt requested. If the vector addresses are eight bytes long, the interrupt controller will automatically supply address bits A0 - A5 and ignore bit 5 in this command word.

Support Circuits

Table 14-3. Initialization Command Word 2

BIT	DESCRIPTION
0-2	These three bits are not used in this computer. Refer to the explanation for the bit 0-7 configuration.
3-7	These five bits contain address bits A3 - A7 in this computer. In computers with 8-bit CPUs, such as the 8080 or 8085, these bits hold address bits A11 through A15. Refer to the explanation for the bit 0-7 configuration.
0-7	This computer does not use this configuration. In computers with an 8-bit CPU, such as the 8080 or 8085, these bits contain address bits A8 - A15.

Table 14-4. Initialization Command Word 3

BIT	DESCRIPTION
0-7	If this interrupt controller is the master device, these bits let the controller know which interrupt request line has a slave interrupt controller attached to it. Each bit corresponds to an interrupt request line.
0-7	If this interrupt controller is a slave device, the three low-order bits contain the binary value that corresponds to the interrupt line of the master interrupt controller attached to this controller. For instance, if this controller is programmed with a 6 (bits 1 and 2 set), then the controller would be connected to interrupt request line 6 of the master interrupt controller. In addition, bit 6 in command word 3 for the master interrupt controller would be set.

Support Circuits

Table 14-5. Initialization Command Word 4

BIT	DESCRIPTION
0	Tells the interrupt controller whether it is in the 8088/8086 (16-bit CPU) mode or the 8080/8085 (8-bit CPU) mode. With this bit set (1), the controller is in the 8086/8088 mode.
1	Tells the controller whether to program for automatic end-of-interrupt mode or not. With this bit set (1), the controller will be in the automatic end-of-interrupt mode. In the automatic end-of-interrupt mode and in the 16-bit mode the interrupt controller will perform a non-specific end-of-interrupt at the trailing edge of the second interrupt acknowledge received from the CPU. In the 8-bit mode, the controller performs this operation at the end of the third interrupt acknowledge received from the CPU.
2	Bit 3 of this word determines whether the interrupt controller is in the master or slave mode. When set (1), the bit indicates that the controller is in master mode. When the bit is clear (0), it will be ignored.
3	This word determines whether the controller is in the buffered mode or not. This controls the output of the $\overline{SP/EN}$ line and the status of bit 2 for this word. If set (1), the buffered mode is selected and the $\overline{SP/EN}$ line becomes an enable output line. Bit 2 of this command word will then be checked for the master/slave option.
4	This word determines whether the interrupt controller works in the fully nested mode or not. In this mode the controller supports more than one level of interrupt and the priority order of the eight lines are arranged from highest (IRQ0) to lowest (IRQ7). If the bit is clear (0), then the controller will not operate in the fully nested mode of operation. In this mode one interrupt will not interrupt the operation of another, but will operate in a sequential manner. Note that the priority order may change under software control.
5-7	Not used.

In this computer, set the interrupt controller as follows:

1. Set the interrupt request inputs, IRQ0 – IRQ15, to be edge-sensitive. The controller will send an interrupt to the CPU on the positive-going transition of one of these lines.
2. Interrupt priority is in reverse numerical order. IRQ0 has the highest priority, IRQ15 the lowest.
3. Clear all interrupt masks.
4. If the interrupt controller sends an interrupt to the CPU, the controller will not send any additional interrupts to the CPU until the CPU executes a specific end-of-interrupt instruction.
5. The interrupt controller uses vector addresses spaced every four bytes in a 32-byte table in memory.

Operation

Operation of the interrupt controller is divided into five areas: CPU mode, priority handling, interrupt trigger, status, and cascading.

CPU Mode — The controller design allows operation with two types of CPUs. The 8259 can operate with eight-bit devices, such as the 8080 and 8085, or sixteen-bit devices, such as the 8088 and 8086. The CPU mode determines the type of device the controller will operate with. The system programs the controller with this information during the initialization command word sequence. In this computer, the interrupt controller always operates in the sixteen-bit CPU mode.

When an interrupt takes place in the sixteen-bit mode, the controller places a single interrupt-vector address byte on the data bus in response to two interrupt acknowledge signals from the CPU.

The controller uses the first signal to resolve interrupt priorities. The second signal times the placing of the interrupt address byte on the data bus. The interrupt request being serviced determines the values for bits 0 – 2. The values for bits 3 – 7 are determined

Support Circuits

by the pattern programmed into the controller during initialization command word 2.

The controller does not use the eight bits placed on the data bus as a direct address. Instead, the value is interpreted as the call number for the INT command, executed by the CPU. These INT commands are the same commands used by software, as explained in Part II of this manual.

When the CPU receives the eight bits from the interrupt controller, the processor multiplies the value by four to obtain the address of the interrupt vector address. Program execution is then transferred to the routine at the interrupt vector address.

Priority handling — Priorities fall into two categories: fully nested and masked. The controller handles them in several different modes. These modes are dynamically programmable; priority handling may be changed during the operation of a program.

The fully nested mode is a general purpose mode that supports multiple levels of interrupt priority. The eight interrupt request lines are handled in a highest-to-lowest manner, usually with interrupt request line 0 as the highest priority. This is the default mode set up during controller initialization.

Programming can change which interrupt request line has the highest priority. For example, you can make interrupt request line 4 the highest priority. Interrupt request line 5 will then be the next highest, interrupt request line 6 the next, and so on. Interrupt request line 3 will be the the lowest priority level.

Once the CPU acknowledges an interrupt, the highest priority request is determined from the interrupt request register. The controller places the vector value on the data bus and sets the corresponding bit in the in-service register. The in-service register bit will remain set until the controller receives an end-of-interrupt command.

In the fully nested mode, once the in-service register bit is set, all subsequent requests by the same or lower-priority interrupt request line will not generate an interrupt to the CPU. If a higher-priority interrupt request occurs, the controller will generate an interrupt.

The higher-priority interrupt will interrupt the execution of the service routine for the lower-priority interrupt. The CPU disables the interrupt request pin when it returns an acknowledge signal to the controller. Because of this, the CPU must have an enable interrupt instruction executed before a higher-priority interrupt request can be acknowledged.

Consider the following example. While the main program is being executed, the in-service register will be clear, since no interrupts are being serviced.

Suppose interrupt request line 3 becomes asserted. The interrupt controller notifies the CPU, which in turn sends an interrupt acknowledge signal. The controller places the vector byte on the data bus and receives the second interrupt acknowledge signal. At that point, the controller sets bit 3 of the in-service register and the CPU starts executing the service routine for interrupt line 3. One of the first instructions executed by this routine is the enable interrupts instruction that allows the CPU to receive further interrupt requests from the controller.

Now, suppose that interrupt line 1 is asserted while the CPU is executing this service routine. Again, the interrupt controller notifies the CPU which sends an interrupt acknowledge signal. The controller places the vector byte on the data bus and receives the second interrupt acknowledge signal. At this point, the controller sets bit 1 of the in-service register and the CPU starts executing the service routine for interrupt line 1. Bit 1 and bit 3 are both set in the in-service register because the service routine for interrupt line 3 was not completed. Also, after the CPU once again receives another enable interrupts command, the controller will act only on interrupt request line 0.

When the service routine for interrupt line 1 has finished, it must inform the controller by executing an end-of-interrupt command. This will clear bit 1 in the in-service register. Next, a return instruction must be executed by the program. This will return control to the service routine for interrupt line 3 and will allow any interrupt line from 0 to 2 to be serviced by the system.

Support Circuits

When the service routine for interrupt line 3 has finished, the end-of-interrupt command will reset bit 3 in the in-service register. When the return instruction executes, control will transfer back to the main program.

The interrupt controller in this computer is almost always in the fully nested mode of operation. Only two programming conditions can disturb this mode: the automatic end-of-interrupt mode and the special mask mode.

Three different end-of-interrupt formats are available through programming. They are the non-specific end-of-interrupt command, the specific end-of-interrupt command, and the automatic end-of-interrupt command.

The non-specific end-of-interrupt command, while letting the controller know that an interrupt service routine was completed, does not inform the controller which level of interrupt was involved. The controller is in a mode where it can determine service routine levels. Because of this, it can determine that the interrupt level that applies to the routine just completed corresponds to the highest interrupt level set in the in-service register. The non-specific end-of-interrupt command will reset this bit.

There are two conditions that may cause the non-specific end-of-interrupt routine to fail. One is when the service routine resets the interrupt priorities. The other is when the special mask mode is in use. In both cases, the controller may not be able to determine the routine's interrupt level.

The specific end-of-interrupt command must include the in-service bit to be reset. This allows the programmer the latitude to change interrupt priorities with the servicing routine or perform other functions that might make it vague to the controller which interrupt routine was being serviced, particularly if other service routines were being executed at the same time.

The automatic end-of-interrupt mode eliminates the need for the CPU to issue an end-of-interrupt command to notify the controller that an interrupt service routine has been completed. While in this

mode, the controller will perform a non-specific end-of-interrupt at the trailing edge of the second interrupt acknowledge signal from the CPU.

This mode disturbs the fully nested mode because the in-service register bit is reset right after it was acknowledged, leaving no sign that the service routine is being executed. Therefore, any interrupt request (when interrupts are enabled) will get serviced, regardless of its priority, making it possible for an interrupt request to interrupt its own service routine.

It is considered good programming practice not to use the automatic end-of-interrupt mode unless the CPU's interrupt input will be kept disabled while interrupt routines are being serviced.

Rotation of the interrupt priorities is available under several conditions and is provided in two modes: automatic rotation and specific rotation.

Automatic rotation is desirable when the interrupts are of equal priority, for example, when a series of communication channels handle peripheral devices. Once a peripheral is serviced, all other equal-priority peripherals should be given the opportunity of being serviced before the first is again serviced. To accomplish this, the automatic rotation mode assigns the just-serviced line the lowest priority. Automatic rotation may be implemented with the rotate on non-specific end-of-interrupt command or in the rotate on automatic end-of-interrupt mode.

When the rotate on non-specific end-of-interrupt command is given, the in-service register bit being serviced is reset and its corresponding interrupt request line is assigned lowest priority. The other lines' priorities are then rotated to conform to the fully nested format, based on the bit that has been assigned the lowest priority. For example, if bit 3 was just serviced, it is assigned lowest priority, bit 4 the highest, bit 5 the second highest, and so on around to bit 2, which is second lowest. The rotate on automatic end-of-interrupt mode works much the same way, except that the reassignment of priorities takes place at the falling edge of the second interrupt acknowledge received from the CPU.

Support Circuits

Specific rotation is completely controllable by the programmer. Through this operation, the specific interrupt request line is selected to receive the highest or lowest priority. Two commands allow this: the set priority command and the rotate on specific end-of-interrupt command.

The set priority command assigns an interrupt request line to the lowest priority. The other lines' priorities are then rotated to conform to the fully nested format, based on the bit that has been assigned the lowest priority.

If the set priority command is used during a service routine, then you must use either a specific end-of-interrupt command or the automatic end-of-interrupt mode to end the routine. The non-specific end-of-interrupt resets the highest in-service register bit, which may not represent the service routine that issued the set priority command. If the automatic end-of-interrupt mode is used, it performs the non-specific end-of-interrupt before the set priority command can be issued; however, the best practice is to use the specific end-of-interrupt command to eliminate any possible confusion.

The rotate on specific end-of-interrupt command is a combination of the set priority command and the specific end-of-interrupt command. With this command, you specify which interrupt request line is assigned to the lowest priority and issue the end-of-interrupt command at the same time. The other lines' priorities are rotated to conform to the fully nested format, based on the bit that has been assigned the lowest priority.

Masking interrupts allows the programmer to enable interrupt request lines that are at a lower priority than the one being serviced. As an example, suppose interrupt request line 4 has triggered its interrupt service routine, but you want to allow lower-level interrupt request lines to interrupt the service routine. Inside the service routine for interrupt request line 4, you would first mask interrupt request line 4 and then issue a special mask mode command. This disables normal nested mode priority operation and enables all interrupt request lines except those being masked. To leave this mode, execute the sequence in reverse order.

There is one problem with using the mask mode. You cannot use a non-specific end-of-interrupt command because all masked inter-

rupt request line bits are hidden and are not clearable from the in-service register. You must exit the mask mode to use the non-specific end-of-interrupt command to clear masked bits from the in-service register. Therefore, it is best to issue a specific end-of-interrupt command when using this mode.

Interrupt trigger — This computer uses two standard methods to sense an interrupt: level-sensitive and edge-sensitive.

In the level-sensitive interrupt mode, the interrupt controller will recognize an active-high on any of its interrupt request lines. If the interrupt request line remains active after the end-of-interrupt command is issued, another interrupt will be generated if the CPU has been told to recognize interrupts. In this mode, the interrupt must remain active until the first interrupt acknowledge from the CPU has been received. Otherwise, the controller will act as if interrupt request line 7 had been active.

The edge-sensitive mode is the default in this computer. In this mode, the interrupt controller recognizes the interrupt on the rising edge as an interrupt request line goes active. If the interrupt request line remains active after the service routine has been completed and the processor is set to recognize interrupts, it will not trigger a subsequent interrupt. In this mode, the interrupt must also remain active until the first interrupt acknowledge from the CPU has been received. Otherwise, the controller will act as if interrupt request line 7 had been active.

Status — There may be occasions where the status of the three internal registers needs to be known, particularly by an interrupt service routine. Polling of interrupts is also possible, though it is not needed in this computer.

The three internal registers of the interrupt controller, the in-service register, the interrupt request register, and the interrupt mask register, can be read by software. The interrupt request register specifies all interrupt request lines that are currently requesting service. The in-service register specifies all interrupt request lines that are currently being serviced by routines. The interrupt mask register specifies all interrupt request lines that are currently masked.

Support Circuits

Cascading — The 8259 can operate in an optional cascade mode. This mode, is used in this computer, and allows a single master controller to control up to eight slave controllers to access 64 different priority levels. This computer uses one master and one slave controller for a total of 15 priority levels. The three cascade lines are used in a manner similar to a chip select signal to select between different slave controllers. In this mode, each controller is initialized separately. Also, at the end of an interrupt cycle, separate end-of-interrupt commands (EOI) must be issued for the master and the slave controller.

Interrupt Controller Port Address

The interrupt controller is treated as an input/output device for programming purposes. Interrupt controller #1 (master) has system address 020H – 03FH. Interrupt controller #2 (slave) has system address 0A0H – 0BFH. If A0 needs to be cleared, use OUT or IN commands to port address 020H. If A0 needs to be set, use OUT or IN commands to port address 021H.

Programming Considerations

The programmable interrupt controller accepts two types of commands: initialization and operation. Normal operation cannot begin until the initialization commands have been issued. Operation commands allow the 8259 to operate in one of four interrupt modes: fully nested mode, rotating priority mode, special mask mode, and polled mode.

Since the system BIOS routines handle both initialization and operation of the 8259, we do not recommend programming the chip directly. For a complete discussion of the interrupt controller and its characteristics, refer to the *iAPX 86, 88 User's Manual*.

Sequence of Operation

Once the interrupt controller has been programmed, the CPU must be instructed to set a bit in its flag register to enable interrupts. If one or more circuits generate an interrupt, the system will respond as follows:

1. One or more of the interrupt request lines, labeled IRQ0 to IRQ7 for interrupt request, will go high.
2. The controller will check the priorities of the incoming interrupts and compare them to others that may also be waiting to be serviced.
3. The controller will assert the INTRQ line to the CPU.
4. The CPU recognizes the interrupt request, completes its current instruction, and places the interrupt acknowledge code on the status bus.
5. The interrupt controller will resolve the priority of the interrupts that are requesting service and prepare the vector address value of the selected interrupt line for placement on the data bus during the next interrupt acknowledge cycle.
6. The CPU generates the second interrupt acknowledge signal.
7. Upon receipt of the second interrupt acknowledge signal from the CPU, the controller places the 8-bit interrupt vector address value on the address/data bus.
8. The CPU multiplies the 8-bit value by four to determine the actual address in the interrupt pointer (vector) table for the interrupt service routine.
9. The CPU also disables interrupts by clearing the flag bit in its flag register. It then pushes the flag register, IP (instruction pointer), and CS (code segment register) values onto the CPU stack.
10. The new IP and CS values are fetched from the interrupt pointer table and control jumps to the routine at that address.

Support Circuits

11. The interrupt-handling subroutine should push any other CPU registers to be used during the service routine onto the stack.
12. Typically, at the end of the service routine, an end-of-interrupt command is sent to the controller. This resets the in-service register bit for the service routine, indicating that the routine has finished.
13. Before the routine is exited, the appropriate CPU registers are returned from the stack, interrupts are enabled, and control returns to the instruction following the one that the CPU completed when the interrupt service routine was called.

There will be variations in this sequence depending upon a number of factors, the least of which is the possibility that alternate modes can be used by the interrupt controller under program control. However, this should provide you an idea of how a typical operation takes place.

Further Programming

Once programmed, the interrupt controller is ready to accept interrupt requests through its input lines. Three different command options allow the controller to operate in different modes. These commands, unlike the initialization command words, do not have to be in sequential order, but may be sent to the controller as needed.

Use command word 1 to place a mask in the controller's interrupt mask register. Each of the data bits sent to the controller during the write operation corresponds to one of the interrupt request lines. Bit 0 corresponds to interrupt request line 0, bit 1 to line 1, and so on. If a bit is set (1), the line is masked. If a bit is clear (0), the line is not masked. Address line 0 must be set prior to executing this write operation to the controller.

Use command word 2 for the end of interrupt and rotation commands. It is summarized in Table 14-6. Address line 0 must be clear (0) for the write operation of this command word.

Table 14-6. Operation Command Word 2

BIT	DESCRIPTION
0 – 2	These three bits, coded in a binary format, determine the interrupt level (request line) that is acted upon by the remainder of this word. If all three bits are clear (0), then the controller programs interrupt request 0. With bit 0 set (1) and bits 1 and 2 clear (0) the controller will program interrupt request 1; and so on. Note that bit 6 of the command word can disable the function of these three bits.
3 – 4	These two bits are clear (0), signifying that this is command word 2.
5	Determines if an end-of-interrupt command is issued by the controller. If the bit is clear (0), the controller will not execute the end-of-interrupt command. If the bit is set (1), bit 6 and bit 7 determine the type of end-of-command operation executed.
6	Enables or disables the use of bits 0 – 2. When set (1), the controller enables the first three bits of the command word. If this bit is clear (0), the controller disables them.
7	Controls all interrupt controller rotation operations. When set (1), the controller will perform rotation, based on the state of bits 6 and 7 in the command word, as follows: <ul style="list-style-type: none"> <li data-bbox="308 866 933 978">● With bits 5 and 6 both set (1), the interrupt request line specified by bits 0 – 2 is set to the lowest priority level. The remainder of the interrupt request lines will conform to the fully nested mode based on this interrupt. <li data-bbox="308 978 923 1010">● With bit 5 set (1) and bit 6 clear (0), bits 0 – 2 are ignored. <li data-bbox="308 1010 933 1090">● When bit 5 and 6 are both clear (0), the controller will not issue the end-of-interrupt command. In this case bits 0 – 2 will be ignored. <li data-bbox="308 1090 912 1117">● If bit 7 is clear (0), the controller will not perform rotation.

Support Circuits

The controller uses command word 3 for a number of purposes. The functions performed by command word 3 are summarized in Table 14-7.

Table 14-7. Operation Command Word 3

BIT	DESCRIPTION
0	This bit determines whether the in-service register or the interrupt request register will be read by a read register status operation. With the bit set (1), the in-service register is read. If this bit is clear (0), the interrupt request register is read.
1	This bit determines whether the controller has issued a read register command. With the bit set (1), the controller issues a read register command. Unless bit 2 is set (1), bit 0 will determine which register is read.
2	This bit determines if the controller has issued a poll command or not. With this bit set (1), the controller overrides bit 1 and issues a poll command. If this bit is clear (0), the controller will not issue the command.
3-4	Bit 3 is set (1) and bit 4 is clear (0), signifying that this is command word 3.
5	This bit determines whether the special mask mode is to be enabled or disabled. Bit 6 controls the validity of this bit.
6	If this bit is set (1), then bit 5 will be honored and the special mask mode will be enabled or disabled, depending on the status of bit 5. If this bit is clear (0), bit 5 will be ignored.
7	Not used.

Pinout

Table 14-8 describes the pin functions of the programmable interrupt controller IC. Refer to Figure 14-2 for an illustration of the pinout.

Support Circuits

Table 14-8. 8259 Interrupt Controller Pin Functions

PIN NUMBER	SIGNAL NAME	DESCRIPTION
1	\overline{CS}	Chip select. This active-low input signal enables read and write communication (programming) between the CPU and the interrupt controller.
2	\overline{WR}	Write. This active-low signal allows the interrupt controller to accept command words from the CPU.
3	\overline{RD}	Read. This active-low signal allows the CPU to read the status of the internal registers of the interrupt controller.
4–11	D0–D7	Data line 0 – data line 7. In this computer, the data lines are connected directly to address/data line 0 – address/data line 7, the bidirectional address/data bus. Command words are sent to, and vector and status data is received from, the interrupt controller over these lines.
12–13	CAS0–CAS1	Cascade line 0 – cascade line 1. These signals are used with pin 15 when more than one interrupt controller is cascaded together, allowing control of more than eight interrupt request lines. NOTE: Do not confuse these signal names with the column address strobe signal lines which are also identified as CAS lines.
14	GND	Ground.
15	CAS2	Cascade line 2. This signal is used with pins 12 and 13 when more than one interrupt controller is cascaded together, allowing control of more than eight interrupt request lines. NOTE: Do not confuse this signal name with the column address strobe signal line 2 which is also identified as CAS2.
16	$\overline{SP/EN}$	Slave program/enable buffer. This line is used in installations where more than one interrupt controller is cascaded together as an input to designate master controller (when tied high) or slave controller (when low). In this type of installation, this line acts as an output to control buffer transceivers under program control.
17	INT	Interrupt. This pin is asserted high when an interrupt request line is recognized and the service routine is requested. It is used to notify the CPU of the request.

Support Circuits

Table 14-8 (continued). 8259 Interrupt Controller Pin Functions

PIN NUMBER	SIGNAL NAME	DESCRIPTION
18 – 25	IRQ0 – IRQ7	Interrupt request line 0 – interrupt request line 7. These asynchronous input lines request an interrupt when they are asserted high and held in that state until the first interrupt acknowledge signal is received from the CPU.
26	INTA	Interrupt acknowledge. This input from the CPU is used to carry the interrupt acknowledge signal to the controller. Two pulses from the CPU are required to complete an interrupt acknowledge sequence.
27	A0	Address line 0. This is used as an input control line, along with the \overline{CS} , \overline{WR} , and \overline{RD} lines.
28	Vcc	+5 VDC power supply.

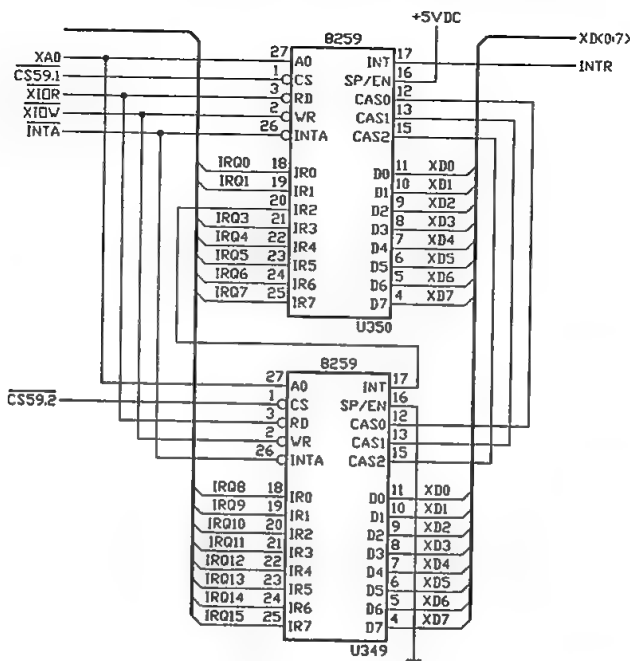


Figure 14-2. 8259 Interrupt Controller Pinout

The 8254 Programmable Interval Timer

The programmable interval timer generates time delays of varying lengths and eliminates the need for software timing loops that are subject to such variables as CPU clock speed and interrupt service routines. Instead, you can configure the timer to your requirements and initialize one or more of the three counters in the timer with the desired values. Then, upon command, the timer will count to the set value and issue an interrupt request upon completion of the task.

The timer also can be used as a programmable rate generator, event counter, binary rate multiplier, real-time clock, digital "one-shot", or complex motor controller. Not all of these functions are used in this computer.

The timer can operate in one of six modes: interrupt on terminal count, programmable one-shot, rate generator, square-wave rate generator, software-triggered strobe, and hardware-triggered strobe. Refer to Figure 14-3, the block diagram of the timer, for the following discussion.

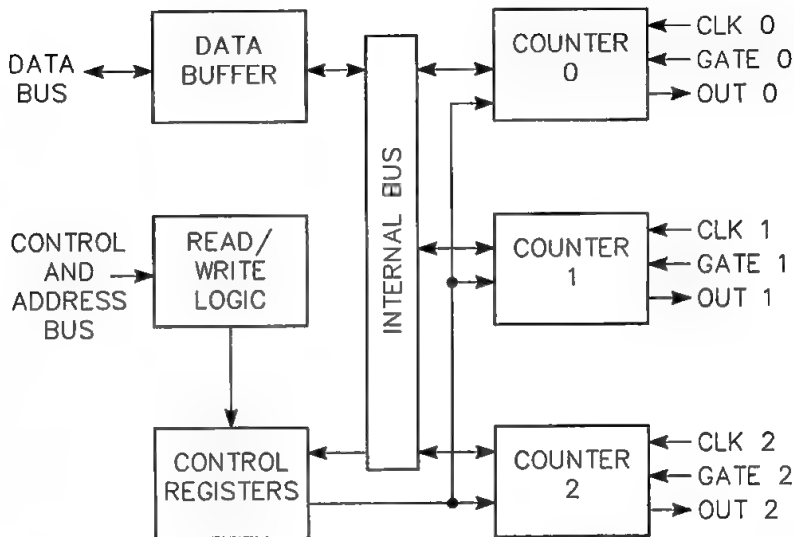


Figure 14-3. 8254 Programmable Interval Timer Block Diagram

Support Circuits

The timer is divided into six logical sections: data buffer, read/write logic, control register, and three counters.

The data buffer is a three-state, bidirectional buffer that interfaces the timer with the address/data bus. Data is sent to or received from the bus via IN or OUT commands to ports 040H – 043H.

The read/write logic accepts the $\overline{\text{IOR}}$, $\overline{\text{IOW}}$, $\overline{\text{CS54}}$, A0, and A1 signal lines as inputs to determine the type of operation to be performed inside the timer. $\overline{\text{CS53}}$ controls chip access, while address lines 0 and 1 determine the type of read or write operation requested.

The control registers receive the instructions that program the counters. The control word for each counter is selected by the value in bits 5 and 6. Table 14-9 describes the contents of the registers. Since all three registers (one for each counter) are identical, only one register is described.

Table 14-9. Timer Control Registers

BIT	DESCRIPTION
0	Establishes the counter format: 16-bit binary or 4-digit binary-coded decimal. If the bit is set (1), the counter will be in the 4-digit binary-coded decimal format. If the bit is clear (0), the counter will be placed in the 16-bit binary format.
1–3	Establish the mode for the selected counter. If these bits equal zero, mode 0 is selected; 1 selects mode 1; 2 or 6 selects mode 2; 3 or 7 selects mode 3; 4 selects mode 4; and 5 selects mode 5. Refer to the text for a description of each mode.
4–5	Control the loading (writing) of the count into the specified counter. If the value placed in these bits is 0, the counters are latched; if the value is 1, only the most-significant byte will be read or loaded; if the value is 2, only the least-significant byte will be read or loaded; if the value is 3, the least-significant byte will be read or loaded first, followed by the most-significant byte. In all cases, the number of bytes (one or two) must be read or loaded before a different operation can be applied to the specified counter.
6–7	Specify the counter to receive the operation. If the value is 0, counter 0 is specified; a value of 1 specifies counter 1; and a value of 2 specifies counter 2. If the value is 3, no counter is selected; the value is illegal.

Each of the three counters contain two bytes and may be configured as either a 16-bit binary counter or a 4-digit binary-coded decimal counter. The input, gate, and output lines are configured by the modes programmed through the control register during programming. In addition, each counter may be read selectively without first inhibiting the clock input for the counter being read.

Mode Definitions

Many of the modes require specific hardware conditions. For example, the gates, clocks, and outputs must be free; not tied to specific circuits. Also, while the programming is fully explained in this section, many of the functions are not available, since the timer is dedicated to specific tasks, as required by PC compatibility. For example, in this computer channel 0 drives interrupt request line 0 (IRQ0) and channel 1 is dedicated to the refresh timing circuits. This means that counters 0 and 1 are not accessible. Channel 2 is available to the programmer. This channel is controlled by bit 0 of port B in the I/O address map (061H – 06FH). The output may be read at bit 5 of this port or it may be used to drive the system speaker but none of the counters can be controlled fully enough to produce special timing signals beyond those allowed by the computer's design.

Interrupt on Terminal Count (Mode 0) — Figure 14-4 illustrates mode 0. This mode decrements the counter until it reaches 0 and then assert its output line high.

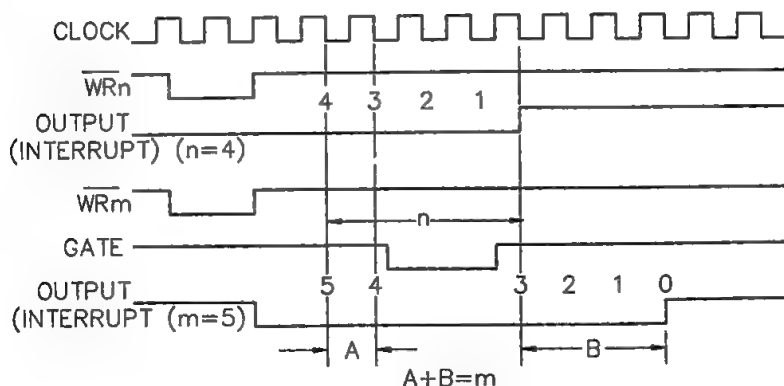


Figure 14-4. Timing Mode 0

Support Circuits

The upper set of lines illustrate the counter not receiving any low signals on the gate input line. Once the counter has been loaded, in this case with a value of 4, the decrementing of the counter starts. When the counter reaches zero, the output will go high. Counting will not resume until the counter is reloaded.

The lower set of lines illustrates the counter being affected by the gate line. In this case, the counter is loaded with a value of 5. As long as the gate line remains high, the counter will decrement. When the gate line goes low, the counter will halt, but not reset to its original value. When the gate line goes high again, the counter will resume decrementing. When the counter reaches zero, the output will go high. Counting will not resume until the counter is reloaded.

In this mode, the output of the counter should be tied to an interrupt line of the CPU.

Programmable One-Shot (Mode 1) — Figure 14-5 illustrates mode 1. In this mode, the counter acts like a programmable one-shot. The output will go low in the clock cycle after the gate is asserted high (acting as a trigger). The output will remain low until the counter has decremented to zero. If the gate input line goes low and then high again, the counter will be reset and will be held low until the counter decrements to zero.

The counter, when acting as a one-shot, is retriggerable. Therefore, the output will remain low until the counter has decremented from the original value to zero following the rising edge of the gate. The rising edge of the gate reloads and reinitiates counting.

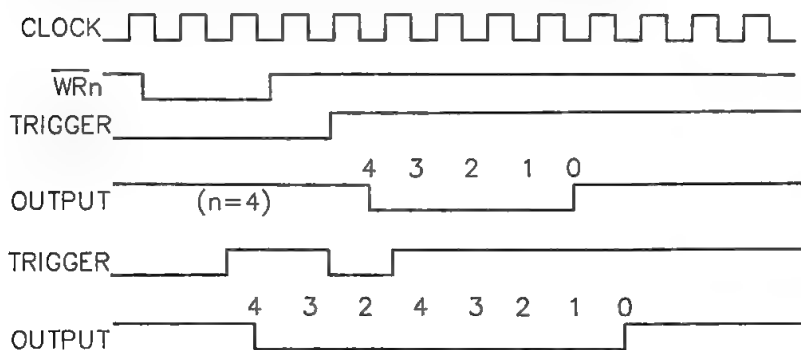


Figure 14-5. Timing Mode 1

Rate Generator (Mode 2) — Figure 14-6 illustrates mode 2. In this mode, the counter will produce a low for one clock cycle on its output pin every n clock cycles. As long as the gate remains high, the counter will repeatedly decrement the value to zero and start over. If the counter's value is changed, the new value will take effect for the next counting cycle and not affect the current cycle. This action is illustrated in the upper set of lines.

The gate can be used to synchronize the counter. While the gate input is low, the counter will not decrement. When the gate goes high, the counter will be reloaded with its programmed value and start decrementing. This action is illustrated in the lower set of lines.

When the gate is low, counting is disabled and the output is made high. The rising edge of the gate reloads and reinitiates counting. When the gate is high, counting is enabled.

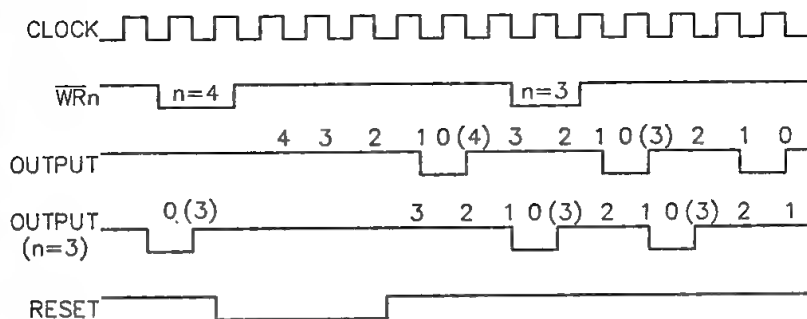


Figure 14-6. Timing Mode 2

Square Wave Rate Generator (Mode 3) — Figure 14-7 illustrates mode 3. In this mode, the counter is decremented by two each clock cycle rather than by one, effectively cutting in half the time it takes to reach zero.

Support Circuits

If the programmed value is even, the counter is always decremented by two. When the counter reaches zero, the state of the output will change (for example, from high to low). The counter is then reloaded and the decrementing by two starts over. When the counter reaches zero, the output state again changes (for example, back to high), the counter is reloaded, and the process starts over again. This produces an even square wave.

If the value is odd and the output is high, the first clock cycle will decrement the counter by one and then by twos until it reaches zero. The clock will then change state to low, the counter will be reloaded with the programmed value, and the first clock pulse will decrement it by three, then by twos until it reaches zero. This produces a signal where the output is high for $(n+1)/2$ counts and low for $(n-1)/2$ counts.

When the gate is low, counting is disabled and the output is high. The rising edge of the gate initiates counting. When the gate is high, counting is enabled.

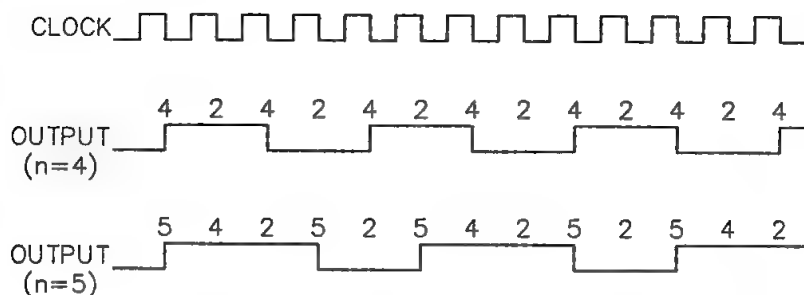


Figure 14-7. Timing Mode 3

Software Triggered Strobe (Mode 4) — Figure 14-8 illustrates mode 4. In this mode, the counter, when it reaches zero, will place a single pulse that lasts for one clock cycle on its output. Counting will begin when the counter is programmed with a value.

When the gate is low, counting is disabled; when the gate is high, counting is enabled.

Support Circuits

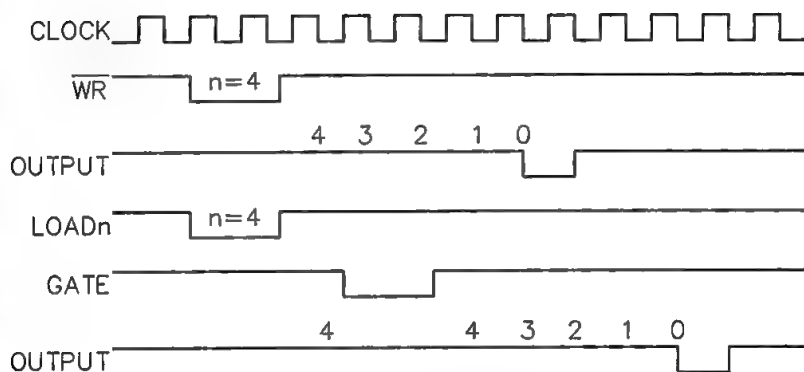


Figure 14-8. Timing Mode 4

Hardware Triggered Strobe (Mode 5) — Figure 14-9 illustrates mode 5. The counter, when it reaches zero in this mode, will place a single pulse that lasts for one clock cycle on its output. The counter will not start decrementing its value until it senses the rising edge of the gate input. The counter is retriggerable and will reload after it reaches zero.

The rising edge of the gate signal initiates counting.

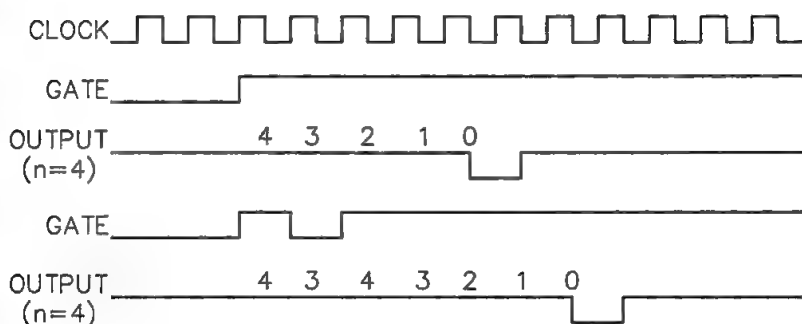


Figure 14-9. Timing Mode 5

Programming Considerations

The Monitor ROM is used to program the mode and initial value in each of the three counters in the timer. In each case a control word is placed in the selected counter's register and then the programmed number of bytes (one or two) before the counter is started.

There is no special sequence for programming the timer. For example, you can write a command word to any of the three counters without affecting the operation or programming of the other two. Suppose you want to load counter 0 with the least-significant byte only, counter 1 with both bytes, and counter 2 with the most-significant byte only. The following two sequences both accomplish this task.

Load Sequence 1:

1. Load command word 0, 1, and 2.
2. Load the most-significant byte in counter 2.
3. Load the least-significant byte in counter 0.
4. Load the least-significant byte in counter 1.
5. Load the most-significant byte in counter 1.

Load Sequence 2:

1. Load command word 2.
2. Load the most-significant byte in counter 2.
3. Load command word 0.
4. Load command word 1.
5. Load the least-significant byte in counter 1.
6. Load the least-significant byte in counter 0.
7. Load the most-significant byte in counter 1.

NOTE: The only sequence that has to be followed is in loading two bytes into a counter. In both examples, the least-significant byte is followed by the most-significant byte before the counter can start operating. In the last example, the least-significant byte and most-significant byte for counter 1 are separated by the least-significant byte for counter 0.

Support Circuits

All counters decrement only; if you load a counter with a value of zero, the actual count will equal the maximum value (10,000 in BCD and 65,536 in binary).

The values in the counters may be read. The address lines determine which counter is read. Normally you would read the contents of the counter with normal read operations. However, you may wish to read the value in the counter while it is operating. This presents problems because you may not get a valid result while the counter is continuing to decrement. The timer offers a method of latching the output of the counter so that it is stable while it is being read. The counter continues to decrement during the read operations but the data remains as it was when it was latched.

The timer can read or write values to the counters through IN (read) and OUT (write) programming instructions. Table 14-10 describes each read and write operation that may be performed on the timer.

Table 14-10. Timer Read and Write Operations

PORT	OPERATION	DESCRIPTION
040H	OUT	Write (load) counter 0.
040H	IN	Read the contents of counter 0.
041H	OUT	Write (load) counter 1.
041H	IN	Read the contents of counter 1.
042H	OUT	Write (load) counter 2.
042H	IN	Read the contents of counter 2.
043H	OUT	If either bit 4 or bit 5 are set (1), write a control word to the specified counter's control register. Refer to Table 14-9 for a description of the contents of the control word.
		If bits 4 and 5 are both clear (0), latch the specified counter's value for reading.
<p>NOTE: The specified counter is identified in bit 6 and bit 7 of the control word.</p>		
043H	IN	Place the data buffer in the high-impedance state; do not perform any operation.

Support Circuits

Pinout

Table 14-11 describes the pin functions of the programmable interrupt controller IC. Refer to Figure 14-10 for an illustration of the pinout.

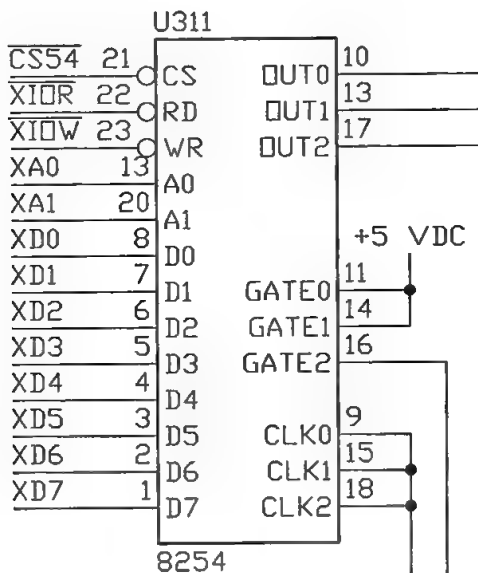
Table 14-11. 8254 Programmable Interval Timer Pin Functions

PIN NUMBER	SIGNAL NAME	DESCRIPTION
1–8	XD7–XD0	Data bit 0 (pin 8) – data bit 7 (pin 1). These pins are three-state bidirectional data lines that carry 8-bit data to and from the timer's counters.
9	CLK0	Clock 0. This line is tied to the CLK1 and CLK2 inputs. All three are driven by the same clock input. It controls the timing for all three counters in the IC.
10	OUT0	Output 0. This line feeds IRQ0 of the interrupt controller to initiate INT 08H.
11	GATE0	Gate 0. This line is tied high so counter 0 always runs.
12	GND	Ground.
13	OUT1	Output 1. This pin feeds the memory refresh timing circuits.
14	GATE1	Gate 1. This line is tied high so counter 1 always runs.
15	CLK1	Clock 1. This line is tied to CLK0 and CLK2.
16	GATE2	Gate 2. This pin allows user control over the programming of the third counter.
17	OUT2	Output 2. This pin is connected to the audio connector to drive the system speaker output.
18	CLK2	Clock 2. This line is tied to CLK0 and CLK1.
19–20	A0–A1	Address line 0 – address line 1. These two lines are used to select the port address of the timer. Refer to Table 14-10 for the ports used by read and write operations. CS is used in conjunction with these lines.
21	$\overline{\text{CS}}$	Chip select. This active-low input line, in conjunction with A0 and A1, determines the port address and actively selects the timer for read and write operations. The system line is CS54.
22	$\overline{\text{RD}}$	Read. This active-low input line will cause the timer to perform a read operation. Refer to Table 14-10 for port addressing.

Support Circuits

Table 14-11 (continued). 8254 Programmable Interval Timer Pin Functions

PIN NUMBER	SIGNAL NAME	DESCRIPTION
23	\overline{WR}	Write. This active-low input line will cause the time to perform a write operation. Refer to Table 14-14 for port addressing.
24	Vcc	+5 VDC power supply.

**Figure 14-10. 8254 Programmable Interval Timer Pinout**

8237 DMA Controller

Direct memory access (DMA) speeds the transfer of data from one memory location, device, or peripheral to another memory location, device, or peripheral. The primary advantage of DMA is in moving blocks of data. The DMA controller has hardware instructions that operate much faster than the software-based CPU instructions. This computer uses two 8237 integrated circuits as the DMA controllers. Refer to Figure 14-11 for a block diagram of the 8237 DMA controller.

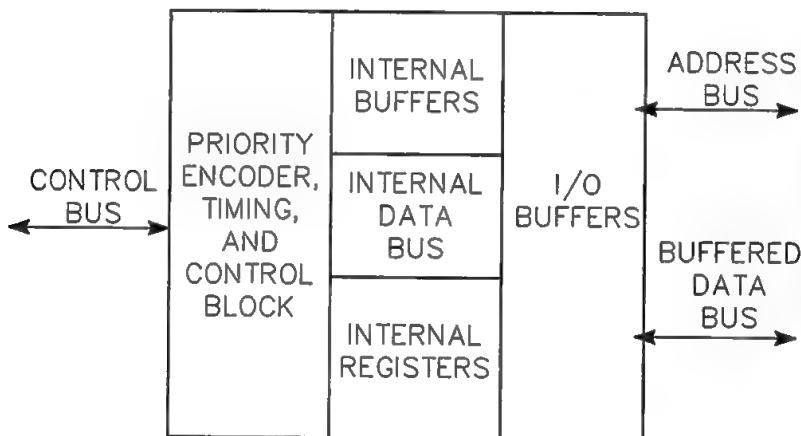


Figure 14-11. 8237 DMA Controller Block Diagram

The DMA controller transfers data using three methods: single byte transfers, defined block transfers, and gated block transfers. The second method is controlled with start and end addresses defined by CPU instructions. The third method is controlled by an external signal.

The controller circuitry operates in one of two types of cycles: the idle cycle or the active cycle. The controller is in an idle cycle when the CPU has control of the system bus. During this time, the CPU can program the controller or read the status registers. The active cycle occurs when the controller has control of the system bus. Some of the input lines become output lines so the controller can send address and control signals to the system bus.

When a transfer of data is necessary, the CPU tells the controller where to get the data and where to put it in memory. The controller then handles the data transfer. It generates up to 16 address lines through address outputs A0-A7. The high-order address is first placed on the address bus and latched. The controller only latches the high-order address when it changes, speeding the transfer process. The address latches and buffers are enabled by a high on the DMA line. A logic array generates all of the memory and input/output read and write signals (MEMR, MEMW, IOR, and IOW) from the S0, S1, and S2 lines.

When the DMA is to control the bus, a hold request signal looks for the CPU idle cycle, as defined by the S0, S1, and S2 lines. When the idle cycle is recognized, the controller locks the CPU in a hold state until the transfer is completed.

A peripheral can request DMA service from the controller on any of the DRQ lines. The controller responds to the input by requesting a CPU hold and then sending a DMA acknowledge output to the peripheral when the CPU enters a hold state. The 8237 design is fully implemented for use in this computer. For example, the DMA controller contains four separate channels, rotating priority logic, and a read buffer.

Internal Registers

In this computer, the DMA controllers are initialized at powerup and zeros are written to the internal register sets. There are eighteen 16-bit registers, three 8-bit registers, four 6-bit registers, and two 4-bit registers inside each DMA controller that store instructions written to the device by the CPU. Refer to Table 14-12 for a description of each register. The following paragraphs describe each of the registers within the 8237 DMA controller.

Support Circuits

Table 14-12. 8237 Internal Registers

REGISTER	SIZE	NUMBER
Base address registers	16-bit	4
Base word count registers	16-bit	4
Current address registers	16-bit	4
Current word count registers	16-bit	4
Temporary address register	16-bit	1
Temporary word count register	16-bit	1
Status register	8-bit	1
Command register	8-bit	1
Temporary register	8-bit	1
Mode registers	6-bit	4
Mask register	4-bit	1
Request register	4-bit	1

Base Registers — These registers include the base-address registers and the base word count registers. There are two of each of these registers for each DMA channel. The values contained in the base registers represent the original value placed in the current address and current word registers (discussed later) by the host processor. The purpose of the base registers is to load the current registers with their base values at initialization.

The base registers cannot be read by the host processor but values can be placed in the registers (simultaneously) when the host processor is in program mode.

Current Registers — There is a current address register and a current word count register for each DMA channel. The current address register reflects the source and destination addresses for data transfers. During a data transfer any intermediate address values are stored in the temporary address register. Each time a data transfer is completed, the current address register is incremented or decremented. The current address register can be reloaded with its original value through execution of an initialization routine. The original value is supplied by the base address register when the EOP (end of process) signal is active.

The current word count register is loaded with a value that determines how many data transfers to execute. When determining this factor, keep in mind that the device will execute one additional transfer beyond the number that it is programmed to execute. For example, if ten transfers are required by the system, the current word count register must be programmed to execute nine operations. After each transfer occurs the register is decremented. Intermediate values during a given transfer are stored in the temporary word count register. The current word count register can be loaded with its original value through execution of an initialization routine. The original value is supplied by the base word count register when the EOP (end of process) signal is active.

The current registers can be read by the host processor and values can be placed in the registers (simultaneously) when the host processor is in program mode.

Status Register — This 8-bit register provides the host processor with current status (with regard to DMA) of devices in its environment. When the processor reads this register, it can determine which channels are programmed to execute an operation and which ones have reached a terminal count (completed an operation). Bits 4 – 7 of the status register indicate that a corresponding channel, if used in the system, is requesting service.

Command Register — The command register contains eight bits that control the entire operation of the DMA controller. The host processor can dynamically place values in this register when the CPU is in program mode. The contents of the register can be cleared through a system reset or a clear instruction implemented through software. Table 14-13 details the functions of each bit in the command register.

Support Circuits

Table 14-13. Command Register

BIT	DESCRIPTION
0	When this bit is held to a zero, memory-to-memory transfers are not allowed. If this bit is set to a one, then memory-to-memory transfers are enabled.
1	When this bit is held to a zero, the channel 0 address-hold register (temporary register) is disabled. If set to a one, then the address-hold register is enabled. Note that if bit 0 is held to a zero, then the status of bit 1 is meaningless.
2	When this bit is held to a zero, the controller is enabled. If this bit is set to a one, the controller is disabled.
3	The status of bit 3 affects the controller's timing mode. If bit 3 is held to a zero, then normal timing will occur. If bit 3 is set to a one, then compressed timing will occur.
4	The status of bit 4 establishes the priority basis on which the controller operates. If bit 4 is held to a zero, then a fixed priority basis occurs. If bit 4 is set to a one, then a rotating priority basis is initiated.
5	The status of bit 5 provides late or extended write selection. If bit 5 is held to a zero, then late write selection occurs. If bit 5 is set to a one, then extended write selection occurs.
6	The status of bit 6 determines the level of the DMA request sense line. If bit 6 is held to a zero, the sense line requires an active-high signal. If bit 6 is set to a one, the sense line requires an active-low signal.
7	The status of bit 7 determines the level of the DMA acknowledge sense line. If bit 7 is held to a zero, the sense line requires an active-low signal. If bit 7 is set to a one, the sense line requires an active-high signal.

Temporary Register — When a memory-to-memory transfer is taking place, this register acts as a holding tank for data. The byte is moved into the register, remains there while addressing and other bus operations are occurring, and then is moved to its specified memory location. After the transfer is completed, the last byte transferred remains in the register unless cleared. The last byte can also be read by the host processor when in program mode.

Mode Registers — The mode register is used to set the operating mode of the DMA controller, provide operation status, and select an active DMA channel. Each DMA channel has its own 6-bit mode register that can be written to by the CPU when it is in program mode. Table 14-14 details the function of each bit in the mode register.

Table 14-14. Mode Register

BIT	DESCRIPTION		
0 – 1	The status of bits 0 and 1 determines which DMA channel is active according to the following logic:		
	BIT 0	BIT 1	
	0	0	Channel 0
	0	1	Channel 1
	1	0	Channel 2
	1	1	Channel 3
2 – 3	The status of bits 2 and 3 determines read, write, or verify transfer status according to the following logic:		
	BIT 2	BIT 3	
	0	0	Verify
	0	1	Write
	1	0	Read
	1	1	Invalid
NOTE: If bits 6 and 7 are both set to one, then the values of bits 2 and 3 are meaningless.			
4	Bit 4 enables the controller's self-initialization routine if it is set to a one. If this bit is held to a zero, then self-initialization is not possible.		
5	Bit 5 causes the value in the current address register to be decremented if set to a one. If this bit is held to a zero, then the address in the current address register will be incremented.		

NOTE: If bits 6 and 7 are both set to one, then the values of bits 2 and 3 are meaningless.

Support Circuits

Table 14-14 (continued). Mode Register

BIT	DESCRIPTION		
6-7	The status of bits 6 and 7 determines the operating mode in which the controller will execute data transfers according to the following logic:		
	BIT 6	BIT 7	
	0	0	Demand mode
	0	1	Single mode
	1	0	Block mode
	1	1	Cascade mode

Mask Register — The mask register can be used to disable DMA requests on each DMA channel. The values placed in the mask register determine which channel is masked or not masked. Table 14-15 details the function of the bits responsible for masking operations.

Table 14-15. Mask Register

BIT	DESCRIPTION		
0-1	These two bits can be programmed to represent four binary logic states. Each condition causes the related DMA channel to be masked according to the following logic:		
	BIT0	BIT1	
	0	0	Channel 0 masked
	0	1	Channel 1 masked
	1	0	Channel 2 masked
	1	1	Channel 3 masked
2	The status of bit 2 either clears each mask bit for each channel or enables the conditions explained for bits 0 and 1. If this bit is held to a zero, all mask bits will be cleared. If this bit is set to a one, the binary combinations of bits 0 and 1 are enabled to mask the chosen register.		

When a master reset occurs (during powerup, for example), each bit for each channel is set to one. DMA requests on each channel will be disabled until a clear-mask instruction changes the state of the appropriate bits. In other words, software can be used exclusively to control the operation of the mask registers. Table 14-16 details how a single command can be used to provide the same masking as detailed in Table 14-15.

Table 14-16. Single Mask Command

BIT	DESCRIPTION
0	If held to a zero, DMA channel 0 will not be masked; if set to a one, channel 0 will be masked.
1	If held to a zero, DMA channel 1 will not be masked; if set to a one, channel 1 will be masked.
2	If held to a zero, DMA channel 2 will not be masked; if set to a one, channel 2 will be masked.
3	If held to a zero, DMA channel 3 will not be masked; if set to a one, channel 3 will be masked.

After completion of a given data transfer, the end of process (\overline{EOP}) signal is asserted by the controller. Unless the device has been programmed to execute its self-initialization routine, each mask register bit will be placed in a set condition. To prevent masking of DMA requests on a given channel after each transfer operation, program the 8237 to initialize itself.

Request Register — The request register is used to prioritize DMA requests. It can respond to requests initiated through software or hardware (DREQ signal). The priority of the requests is established by an internal encoder network. One request bit that cannot be masked is present for each DMA channel. Each of these bits can be set under software control. Table 14-17 details the function of each bit in the request register.

Support Circuits

Table 14-17. Request Register

BIT	DESCRIPTION		
0-1	These two bits can be programmed to represent four binary logic states. Each condition causes the request bit for each DMA channel to be set according to the following logic:		
	BIT 0	BIT 1	
	0	0	Set channel 0
	0	1	Set channel 1
	1	0	Set channel 2
	1	1	Set channel 3
2	The status of bit 2 either clears each request bit for each channel or enables the conditions explained for bits 0 and 1. If this bit is held to a zero, all request bits will be cleared. If this bit is set to a one, the binary combinations of bits 0 and 1 are enabled to set the request bit in the chosen register.		

Operation

The DMA controller operates in either an idle cycle or an active cycle, depending on the status of the devices in the system that are requesting service. If no service is being requested, the DMA controller is idle. It assumes the active state when service has been requested.

When the controller is idle there is no valid software- or hardware-generated DMA request on any of the request lines. The controller continually samples each request line until one of them asserts an active request for service. If no requests occur and the CPU is free, the CPU can dynamically program the DMA controller's registers. The 8237 chip select signal and the hold acknowledge signal from the CPU must both be low to place the controller in program mode.

The CPU can change values in one of the registers or merely read the status of the registers. A 4-bit address applied across lines A0 - A3 acts as a pointer to the desired register. Values can be read or written in a particular register in accordance with the IOR (input/output read) and IOW (input/output write) control signals.

As long as no devices in the system (or software) assert active DMA request signals, the DMA controller will continue to inspect the system and accept programming information from the CPU. However, when an active request for DMA service occurs on a channel that has not been previously masked, the DMA controller enters its active cycle.

When the 8237 enters the active cycle the first procedure that it executes is a hold request (HRQ) to the system CPU. The request will be asserted until the processor recognizes it, completes its current operations according to priority, and then issues a hold acknowledge (HLDA) back to the DMA controller. This acknowledgment informs the controller that the system busses are at its disposal.

The 8237 DMA controller can execute data transfers in four different modes of operation. Depending on the specific hardware environment, different modes may be used to provide excellent system throughput and performance. The modes are described in the following paragraphs.

Single Transfer Mode — In this mode of operation, the 8237 will execute one data transfer. The value in the current word count register will be decremented upon completion of the transfer and the value in the current address register may be incremented or decremented (depending on system requirements) upon completion of a transfer. If the current word count register increments beyond the value of FFFFH, a terminal count indication will occur and the controller will initialize itself (if programmed to). Self-initialization is discussed in the "Special Features" section.

During the cycle, the channel asserting the request signal must continue to assert it until the CPU issues a hold acknowledge to the 8237, which then issues a DMA cycle acknowledgement (DACK) to the requesting device. When the transfer is completed, the system busses are relinquished even if the request for DMA service is still active. The cycle would then repeat.

Support Circuits

Block Transfer Mode — In this mode of operation, the requesting device must wait for acknowledgement from the controller, which must wait for acknowledgement from the CPU, before the system busses will be available. Once the transfer starts, service will continue until a terminal count indication is generated. The terminal count occurs when the value in the current-word-count register exceeds FFFFH. Service may also be stopped if the end-of-process signal is asserted. When the transfer stops, the controller will self-initialize (if programmed to) as discussed in the "Special Features" section.

Demand Transfer Mode — In this mode, transfers will continue unless a terminal count (caused by the value in the current word count register exceeding FFFFH) or end-of-process signal is generated or the requesting channel is masked or goes inactive. An advantage of this mode is that, if a transfer is made from memory to an I/O device that contains some type of data buffer, the transfer continues until that buffer is full and the request for service goes inactive. The system processor then executes other operations until the I/O device's buffer empties and it issues another request for service. Any values that are pertinent to the operation are preserved in the current address and current word count registers until the operation resumes. In this mode the controller will not self-initialize unless it receives an end of process signal.

Cascade Mode — The cascade mode allows more than one 8237 DMA controller to be connected to a master controller. The slaves have to issue their hold request signals to the master controller. The hold-request is issued over the master's DMA request channel and the master's hold acknowledge signal is distributed to the slaves over the DMA acknowledge channel.

The master maintains direct communication with the system CPU to allow the CPU to establish programming values and to allow itself to ascertain when the system busses are free. One advantage of cascading controllers is that the slave controllers provide a type of prioritization over incoming requests for service. If multiple devices are requesting service, the latter of those devices must wait for acknowledgement in a prioritized manner.

Special Features

There are certain features of the 8237 DMA controller that can be used to increase overall system performance. These features include a memory-to-memory data transfer mode, a self-initialization process, a priority management feature, and a compressed timing mode.

Memory-To-Memory Transfers — The 8237 DMA controller can perform transfers from one area of memory to a different area of memory. If bit 0 in the command register is set to a one, DMA channels 0 and 1 are programmed to operate as an interfacing channel between one segment of memory and another segment of memory.

The service request for memory-to-memory transfers must occur through software because the normal DREQ (DMA request) lines are not tied directly to memory. The request and acknowledge logic is managed as with a typical hardware request, however.

During the actual transfer, the controller operates in its block transfer mode. The current address register for channel 0 provides the addresses of the memory string to be transferred. As the operation takes place, this register is incremented or decremented as required.

The bytes of data that are fetched during the operation are temporarily held in the controller's temporary register. The current address register for channel 1 specifies the address at which the data byte is to be placed. This value is incremented or decremented as required. The data byte is then passed through channel 1 to the target memory location.

The channel 1 current word count register is decremented until it counts beyond 0000H (rolls under to FFFFH). It then generates a terminal count which causes an end of process signal to be issued to the system. External end of process signals will also terminate the operation.

Support Circuits

Self-Initialization — When bit 4 of the mode register is set to a one, it allows the controller to be placed in the self-initialization mode. During normal transfers, values in the current address register and current word count register may be changed. When an operation is completed, these registers contain their last hexadecimal value. Self-initialization clears these registers and then loads them with default values.

The base address register and base word count register contain the default values that are loaded into the current address register and current word count register. This process occurs after detection of an end of process signal. After self-initialization, DMA operation can occur without intervention whenever a valid request is made.

Priority Management — The 8237 DMA controller can prioritize DMA service on a fixed priority basis or a rotating priority basis. These options can be selected through software.

Fixed priority simply means that channel 0 has priority over channel 1 and that channel 3 has lower priority than channel 2 if no requests for service have been made. Whenever a request is serviced and an operation is occurring through a given channel, that channel has priority over any other channels. When the operation is completed, channel 0 will be checked for a service request, then channel 1, and so on.

Rotating priority means that the channel currently performing an operation will have the lowest priority after the operation is completed. The advantage to rotating priority is that each channel receives service at least once every fourth service request. This provides a well-balanced environment for DMA operations and prevents prolonged service through one specific channel.

Compressed Timing — The compressed timing mode allows greater throughput in those systems that can support it. Basically, the controller executes the same operation in only two clock cycles. This mode should not be used if system timing requirements are such that bus contention will occur or data set-up times will not be met.

Programming Considerations

The 8237 DMA controller may be dynamically programmed by the CPU whenever the HLDA line is held inactive. Even if a request for the system busses is being made by the 8237, the CPU can still provide instructions to the controller. The CPU must not attempt to program the DMA controller whenever HLDA is active.

NOTE: If a request for DMA service occurs over an unmasked request line while the CPU is attempting to specify a particular address, that address may be corrupted. Then, when the operation is carried out, the DMA controller will attempt to address the wrong memory location, and therefore transfer the wrong byte of data. Therefore, it is good practice to mask each of the request lines before you attempt to dynamically program the 8237 DMA controller.

Pinout

Table 14-18 describes the function of each pin in the 8237 DMA controller. Refer to Figure 14-12 for the pin locations.

Table 14-18. 8237 Pin Functions

PIN NUMBER	SIGNAL NAME	DESCRIPTION
1	$\overline{\text{IOR}}$	Input/output read. When this signal is asserted, it instructs the controller that a read operation over the I/O data bus is to occur.
2	$\overline{\text{IOW}}$	Input/output write. When this signal is asserted, it instructs the controller that a write operation over the I/O data bus is to occur.
3	MEMR	Memory read. When this signal is asserted, it instructs the controller that a read operation from memory is to occur.
4	$\overline{\text{MEMW}}$	Memory write. When this signal is asserted, it instructs the controller that a write operation to memory is to occur.

Support Circuits

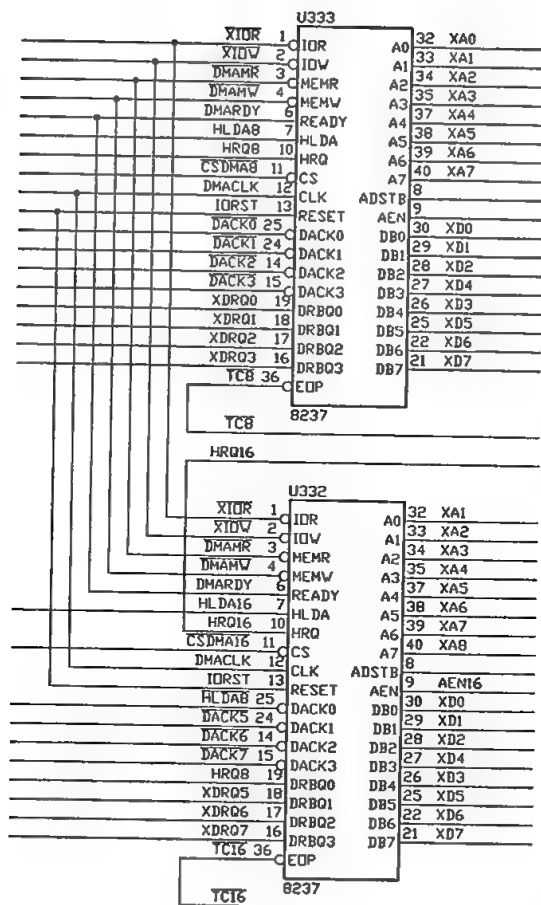
Table 14-18 (continued). 8237 Pin Functions

PIN NUMBER	SIGNAL NAME	DESCRIPTION
5	—	Not used.
6	READY	This signal is asserted to accommodate slow memory accesses or I/O operations. It causes extra time during these operations to make sure that data has a valid set-up time.
7	HLDA	Hold acknowledge. This signal is issued by the CPU in response to a hold request by the 8237. When asserted, the DMA controller can take over the system busses to complete a data transfer.
8	ADSTB	Address strobe. On a low-to-high transition, this signal causes the upper address byte to be transferred into its appropriate latch.
9	AEN	Address enable. This signal allows the eight upper bits of the address bus to be placed onto the system address bus. It is also used for bus arbitration during transfers.
10	HRQ	Hold request. This signal is asserted by the 8237 to request control of the system busses. When the CPU recognizes the request, it issues a hold acknowledge to the 8237.
11	$\overline{\text{CS}}$	Chip select. When the chip select signal is active-low, it enables the DMA controller to be used as an I/O device to handle data transfers.
12	CLK	Clock. This signal controls the operation of the internal register structure and determines the rate at which data transfers occur.

Table 14-18 (continued). 8237 Pin Functions

PIN NUMBER	SIGNAL NAME	DESCRIPTION
13	RESET	This active-high signal causes the status of all internal registers of the 8237 to be cleared. After an active RESET, the controller enters an idle state.
14	DACK2	DMA acknowledge channel 2.
15	DACK3	DMA acknowledge channel 3.
16-19	DREQ3-DREQ0	DMA request channel 3 – DMA request channel 0.
20	GND	Ground.
21-23	DB7 – DB5	Data bus. Data bit 7 – data bit 5.
24	DACK1	DMA acknowledge channel 1.
25	DACK0	DMA acknowledge channel 0.
26-30	DB4 – DB0	Data bus. Data bit 4 – data bit 0.
31	VCC	+5 VDC.
32-35	A0 – A3	Address bus. Address bit 0 – address bit 3.
37-40	A4-A7	Address bus. Address bit 4 – address bit 3.
36	$\overline{\text{EOP}}$	End of process. This signal indicates completion of a DMA data transfer.

Support Circuits



The MC146818A Real-Time Clock

The Advanced Desktop Computer Series includes an internal real-time clock to maintain the system time and date. The MC146818A is the device that controls these operations. Unlike the other support circuits discussed in this chapter, the real-time clock is located on the backplane board. This device includes several useful features:

- A time-of-day clock with alarm and 100-year calendar
- Programmable periodic interrupt
- 50 bytes of on-chip static RAM
- Automatic end-of-month and leap year adjustments
- Daylight saving time option

Complete technical specifications for this device, including operating characteristics and timing diagrams, may be found in the *Motorola Microprocessors Manual*, published by Motorola. The following material is a summary of that information.

Architecture

The MC146818A is a peripheral device used to keep track of real time. Although the device has a number of other capabilities (refer to the manufacturer's specifications), in this computer it serves as the system time piece. It records and tracks the current date and time even when the computer is turned off. It also automatically tracks the end-of-month, leap years, and daylight saving time. The internal 100-year calendar ensures accurate date recording for the life of your computer system. Figure 14-13 is a block diagram of the MC146818A real-time clock.

Support Circuits

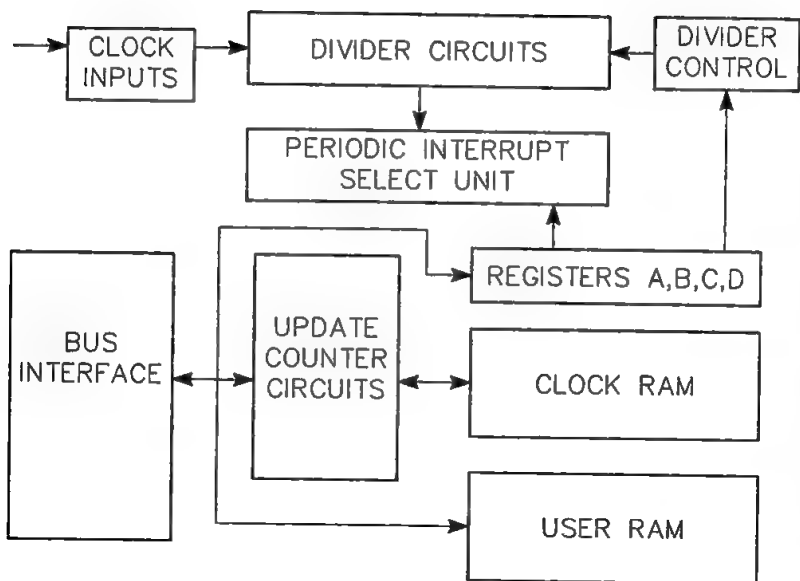


Figure 14-13. MC146818A Real-Time Clock

Internal Registers

Most of the MC146818A consists of 14 bytes used as registers to record timekeeping information and 50 bytes of RAM. In addition to the memory storage areas, the device incorporates a binary divider to control interrupt selection and a bus interface to move data to and from the system bus lines. Figure 14-14 illustrates the internal memory map of this device.

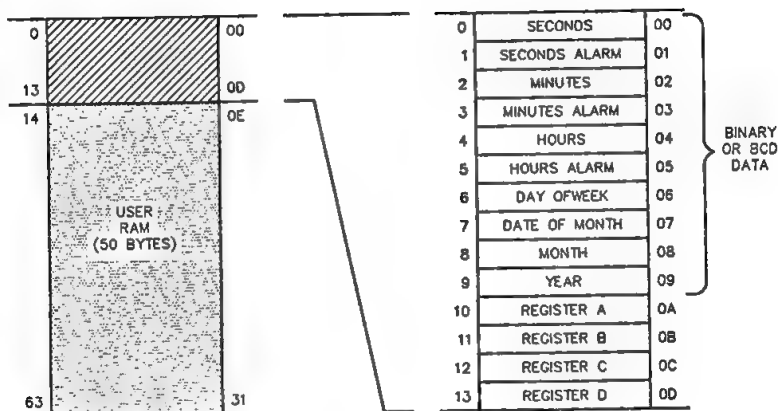


Figure 14-14. Internal Address Map

Of the 14 bytes used for timekeeping information, the first 10 bytes contain specific time-related information. The remaining four registers are used to record status information for the device. The status bytes are not totally accessible to programmers. Registers C and D are read-only registers. Bit 7 in register A is also read-only. In addition, the high-order bit of the seconds byte is a read-only bit. Figure 14-15 details the organization of the four internal status registers. Refer to this figure as you read the following material.

	MSB				LSB				
	7	6	5	4	3	2	1	0	BIT #
REGISTER A	UIP	DV2	DV1	DV0	RS3	RS2	RS1	RS0	READ/WRITE
REGISTER B	SET	PIE	AIE	UIE	SQWE	DM	24/12	DSE	READ/WRITE
REGISTER C	IRQF	PF	AF	UF	READ ONLY
REGISTER D	VRT	READ ONLY

Figure 14-15. MC146818A Status Registers

Support Circuits

Register A

Each bit in register A can be read or written with the exception of bit 7 which is a read-only bit. Table 14-19 describes the functions of the bits in this register.

Table 14-19. Register A Bit Operations

BIT	NAME	DESCRIPTION
0-3	RS0-RS3	Rate selection bits 0 through 3. These bits allow the programmer to select one of 15 separate taps on the internal 22-stage divider. This establishes the rate for the periodic interrupt cycle. (Table 14-20 details these selection options.)
4-6	DV0-DV2	Divider selection bits 0 through 2. The programmer uses these bits to reset the internal divider chain in the MC146818A.
7	UIP	Update in progress bit. When this status flag is set (1), an update cycle is in progress or due to begin. When this bit is clear (0), the time, calendar, and alarm information is available to the programmer. If the bit is clear, another update cycle will not start for at least 244 μ s.

Table 14-20. Periodic Interrupt Time Interval Options

RS0	RS1	RS2	RS3	RATE
0	0	0	0	None
0	0	0	1	3.90625 ms
0	0	1	0	7.8125 ms
0	0	1	1	122.070 μ s
0	1	0	0	244.141 μ s
0	1	0	1	488.281 μ s
0	1	1	0	976.562 μ s
0	1	1	1	1.953125 ms
1	0	0	0	3.90625 ms
1	0	0	1	7.8125 ms
1	0	1	0	15.625 ms
1	0	1	1	31.25 ms
1	1	0	0	62.5ms
1	1	0	1	125 ms
1	1	1	0	250 ms
1	1	1	1	500 ms

Register B

All of the bits in register B can be read or written by the programmer. Table 14-21 describes the functions of the bits in this register.

Table 14-21. Register B Bit Operations

BIT	NAME	DESCRIPTION
0	DSE	Daylight saving time bit. If this bit is set (1), then the daylight saving time option is enabled. This will cause special updates to occur in April and October of each year.
1	24/12	24 hour/12 hour mode select. If this bit is set (1) then the 24 hour mode selection is active.
2	DM	Data mode bit. This bit determines whether updates are to occur in binary or BCD format. A 1 in this bit sets the mode for binary updates.
3	SQWE	Square-wave enable bit. In systems where this signal is connected, a set bit will produce a square wave output at the frequency selected by the rate selection bits in register A.
4	UIE	Update-ended interrupt enable bit. If set, this bit allows the update-end flag bit in register C to assert $\overline{\text{IRQ}}$.
5	AIE	Alarm enable interrupt bit. If this bit is set, the alarm flag bit in register C can assert the $\overline{\text{IRQ}}$ line.
6	PIE	Periodic interrupt enable bit. If set, this bit allows the periodic interrupt flag in register C to assert the $\overline{\text{IRQ}}$ line. The rate of this interrupt is determined by the rate selection bits in register A.
7	SET	When this bit is clear, the update cycle will advance the register counts once per second.

Support Circuits

Register C

Register C is one of two read-only registers in this set. The information in register C is returned by the MC146818A so that a program can obtain current status. Table 14-22 describes the bit functions in this register.

Table 14-22. Register C Bit Operations.

BIT	NAME	DESCRIPTION
0-3	—	Used.
4	UF	Update-ended interrupt flag. This flag is set at the end of each update cycle. If the UIE bit in register B is also set, the IRQF bit will set, asserting the IRQ line.
5	AF	Alarm interrupt flag. If this bit is set at the end of a cycle, it means that the current time matches the stored alarm time. If the AIE bit in register B is also set, IRQF will set, asserting the IRQ line.
6	PF	Periodic interrupt flag. When this bit is set (1), the IRQF bit will also be set if PIE is set.
7	IRQF	Interrupt request flag. This bit is set (1) if one or more of the following conditions are true: PF and PIE are set, AF and AIE are set, or UF and UIE are set.

Register D

Register D is the second read-only register in the MC146818A. Only bit 7, the valid RAM and time (VRT) bit is used. This bit reports the status of the 50 bytes of RAM within the MC146818A. As long as this bit is set, the contents of the onboard RAM and clock registers is valid. If this bit is clear, it indicates that power has dropped below a nominal level. If this happens, assume that the contents of internal clock registers and RAM are no longer valid. Whenever register D is read, bit 7 will be set

Time, Calendar, and Alarm Registers

Ten registers store information about the current time, date, and alarm status in the MC146818A. These registers may be initialized or read by a program except when an update cycle is in progress. Table 14-23 details the available data modes for these registers.

Table 14-23. Time, Calendar, and Alarm Data Modes

REGISTER	FUNCTION	DECIMAL RANGE	RANGES	
			BINARY DATA	BCD DATA
0	Seconds	0-59	00-3B	00-59
1	Seconds alarm	0-59	00-3B	00-59
2	Minutes	0-59	00-3B	00-59
3	Minutes alarm	0-59	00-3B	00-59
4	Hours: 12 hour format	1-12	01-0C 81-8C	01-12 (A.M.) 81-92 (P.M.)
	24 hour format	0-23	00-17	00-23
5	Hours alarm: 12 hour format	1-12	01-0C 81-8C	01-12 (A.M.) 81-92 (P.M.)
	24 hour format	0-23	00-17	00-23
6	Week day	1-7	01-07	01-07 (Sunday = 1)
7	Month date	1-31	01-1F	01-31
8	Month	1-12	01-0C	01-12
9	Year	0-99	00-63	00-99

Operation

The final output of the 22-stage binary-divider is a 1 Hz signal used to update the time keeping information. Once each second all ten bytes are switched to the update logic. When this happens a set sequence of events starts. First, the seconds byte is incremented and then checked for overflow. Next, the minutes byte is incremented, if necessary, and then tested for overflow. This process of incrementing a byte where required and testing for an overflow condition continues through the last timekeeping register. If a program attempts to read this information during an update cycle, it will receive undefined data.

Whenever an update cycle is in progress, the MC146818A switches the timekeeping registers off the system bus to prevent the system from reading transitional data. At the end of this process each alarm byte is compared with its corresponding time byte. If all three match, the MC146818A issues an alarm.

Programming

All the information necessary for programming the MC146818A is contained in the device data sheets from the manufacturer. The device can be accessed at memory address locations 070H and 071H. Address 070H is accessed first to select the desired register in the real-time clock. **The most-significant bit of this address is the non-maskable interrupt bit. Be careful that you do not inadvertently disable this interrupt.** Once the register has been selected, access the location at 071H to perform the read or write operation. The information in the following paragraphs highlights some of the more important points.

Initialization

When first attempting to program the device, check the status of the VRT bit in register D. If this bit is set, the data in the clock registers is valid and may be used immediately. If the bit is clear, you will have to initialize the device.

Start initialization by setting the SET bit in register B. This will disable updates while you are entering initial values into the clock registers. After setting your time options, formats, and initial values, (all using the same data format), be sure to read register D prior to exiting your routine. This will set the VRT bit in that register, indicating that the current information is valid. Your final step should be to clear the SET bit in register B, enabling updates again. Updates will begin one-half second after this bit is clear.

Once the device is initialized, it will perform updates according to the selected data mode. It is not possible to change the data mode without re-initializing the 10 data bytes of clock information.

The clock alarm bytes may be set several different ways. The following is a summary of these modes:

- Setting the alarm bytes to a specific time will cause a recurring alarm at that time each day.

- If a “don’t care” condition (the two most-significant bits of the byte are set) exists in the hours byte, an alarm interrupt results each hour.
- If both the hour and minutes bytes contain a “don’t care” condition, an alarm interrupt will occur each minute.
- If all three time bytes (hour, minute, and second) are set to “don’t care”, the alarm interrupt repeats each second.

Interrupts

The MC146818A has three separate interrupt sources available to the system. The real-time clock can generate an alarm interrupt, a periodic interrupt, or an update-ended interrupt. The control for these interrupts resides in three bits in register B, the AIE bit, the PIE bit, and the UIE bit.

The alarm interrupt enable bit (AIE) controls the alarm associated with the normal clock functions. When the three time bytes match the three alarm bytes and the AIE bit is set, an alarm interrupt will occur each second the condition is true. Setting the AIE bit allows the AF flag in register C to toggle when this condition is true, asserting the device $\overline{\text{IRQ}}$ line. An active condition on the device $\overline{\text{RESET}}$ line will clear the AIE bit.

The periodic interrupt enable bit (PIE) works with the PF bit in register C and the rate selection bits (RS0 – RS3) in register A. When enabled, this bit activates a recurring interrupt. The repetition rate is determined by the value selected by the rate selection bits. When the bit is set, the PF flag will toggle, asserting the $\overline{\text{IRQ}}$ line at the selected rate. Clearing the PIE bit prevents the $\overline{\text{IRQ}}$ line from becoming active but the PF flag will still toggle at the selected rate. Like the AIE bit, the PIE bit will be cleared when the $\overline{\text{RESET}}$ line is active.

The update-enabled interrupt bit (UIE) is the last of the three interrupts. If this bit is set, it enables the UF flag in register C, asserting the device $\overline{\text{IRQ}}$ line. This bit is set by the MC146818A at the end of each update cycle. This bit may be cleared by activating the $\overline{\text{RESET}}$ line or by setting the SET bit in register B.

Support Circuits

Each of the interrupts above operates independently of the other interrupt bits in the B register, but they are associated with their corresponding flag bits in register C. Each interrupt may operate in the flag mode or the fully enabled mode. The difference in their operation relates to the activation of the IRQ line.

Operation in the fully enabled mode is essentially as described above. An interrupt event occurs, the flag bit toggles, and, if the interrupt bit is set, the $\overline{\text{IRQ}}$ line for the device goes active. Flag mode alters this operating sequence slightly. In this mode the $\overline{\text{IRQ}}$ line is not allowed to become active. In either case, the flag bit will toggle as long as an interrupt event occurs, regardless of the setting of the enable bit. This means that a program can monitor the flag bits without actively causing an interrupt request. In this case, the flag bit becomes a status bit as far as software is concerned.

One precaution is necessary when working with the interrupts. Whenever register C is read, all flag bits are cleared. If more than one interrupt is pending, it could be lost. Any program that has more than one interrupt enabled should check all the flag bits. If a program needs to know if the real-time clock generated an interrupt, it should check the IRQF bit in register C. This bit is set whenever the $\overline{\text{IRQ}}$ line goes active. Like the other flag bits in this register, a read of register C will clear this bit.

Update Cycle

The update cycle for the MC146818A occurs once each second and lasts for 1984 μs . During this time, the data in the timekeeping registers is not available to the system. It is possible for a program which randomly reads the time and date information to find that this data is not available. Three separate methods are recommended to handle this situation.

One method uses the update-ended interrupt. This interrupt goes active at the end of each update cycle, if it is enabled. When this bit is set, approximately 999 ms are available in which to read the time and date. A procedure that uses this method should clear the IRQF bit in register C before leaving the interrupt service routine.

Another method takes advantage of the update-in-progress bit (UIP) in register A. The UIP bit will go active whenever an update cycle is in progress. After this bit is set, the next update cycle will begin in 244 μ s. If a program finds that this bit is clear, then a minimum of 244 μ s are available to read data. If the bit is set, the time and date information is not valid.

The final method uses the periodic interrupt to determine if an update cycle is in progress. If a periodic interrupt occurs at a rate greater than once each 2228 μ s, sufficient time is available to read time and date information at each periodic interrupt. The read must be completed within a set time factor. This time factor is the periodic interrupt time interval (refer to Table 14-20) plus 244 μ s. If this factor is exceeded, the data will not be valid.

Memory

The MC146818A contains 50 bytes of static RAM that are available to the system. In this computer, that memory is reserved for use by the Monitor ROM Setup/Configuration program.

The Setup/Configuration program configures the computer to your requirements. (Refer to Chapter 5 for specific details.) The information is retained in the static RAM in the MC146818A so that it can be recalled each time the computer is powered-up.

CAUTION

If you attempt to access this memory, you could unintentionally erase memory information or cause alterations that could damage the computer. At the very minimum, you may change the configuration so that you will need to re-run the SETUP program before you can use the computer again.

Support Circuits

Pinout

Table 14-24 describes the pin functions of the real-time clock IC. Refer to Figure 14-16 for an illustration of the pinout.

Table 14-24. Real-Time Clock Pin Functions

PIN NUMBER	SIGNAL NAME	DESCRIPTION
1	MOT	Motel. This pin allows the device to use different types of bus systems. When connected to +5 VDC the device uses Motorola timing. If this line is connected to ground, alternate timing may be used.
2	OSC1	Oscillator 1. This line is used when the device is connected to an external time base. Pin 3 (OSC 2) is left open in this case.
3	OSC2	Oscillator 2. This line is used with pin 2 to connect the device to an external crystal oscillator circuit. This is the configuration used in this computer.
4–11	AD0–AD7	Address/data lines. This is the multiplexed, bidirectional address/data bus. Address information presented on lines AD0 – AD5 is latched just prior to the fall of AS/ALE. Write data is present during the latter portion of the DS or \overline{WR} cycles. Data to be read is present during the latter portion of the DS or \overline{RD} cycles.
12	VSS	+5 VDC supply voltage input.
13	\overline{CS}	Chip select. This signal must be active if the device needs to be accessed.
14	AS	Multiplexed address strobe. A positive going transition on this line will demultiplex the bus. The falling edge of this signal latches the address on the bus.
15	R/\overline{W}	Read/write. The operation of this signal depends on the condition of the MOT line. In this computer the line is handled in the same fashion as a write pulse for a memory device.

Support Circuits

Table 14-24 (continued). Real-Time Clock Pin Functions

PIN NUMBER	SIGNAL NAME	DESCRIPTION
16	$\overline{\text{STBY}}$	Stand-by. When this pin goes low, the MC146818 cannot be accessed. This will place the device in stand-by mode allowing battery backup.
17	DS	Data strobe. The function of this pin also depends on the status of the MOT line. In this computer the signal is interpreted as a read signal.
18	$\overline{\text{RESET}}$	Reset. This signal is required to insure a stable power supply for the device. It does not affect the operation of the clock, calendar, or RAM.
19	IRQ	Interrupt request. As long as the status bit causing the interrupt is present while the interrupt enable bit is set, this line will remain active. It may be cleared in one of two ways. A RESET signal will clear pending interrupts. If register C is read this will also clear the interrupt.
20	CFKS	Clock out frequency select. This signal can be tied to either +5 VDC or ground. When tied to +5 VDC, as in this computer, it causes the CKOUT signal to have the same frequency as the OSC1 signal.
21	CKOUT	Clock out. Not used in this computer.
22	PS	Power sense. Not used in this computer.
23	SQW	Square wave. Not used in this computer.
24	GND	Ground.

Support Circuits

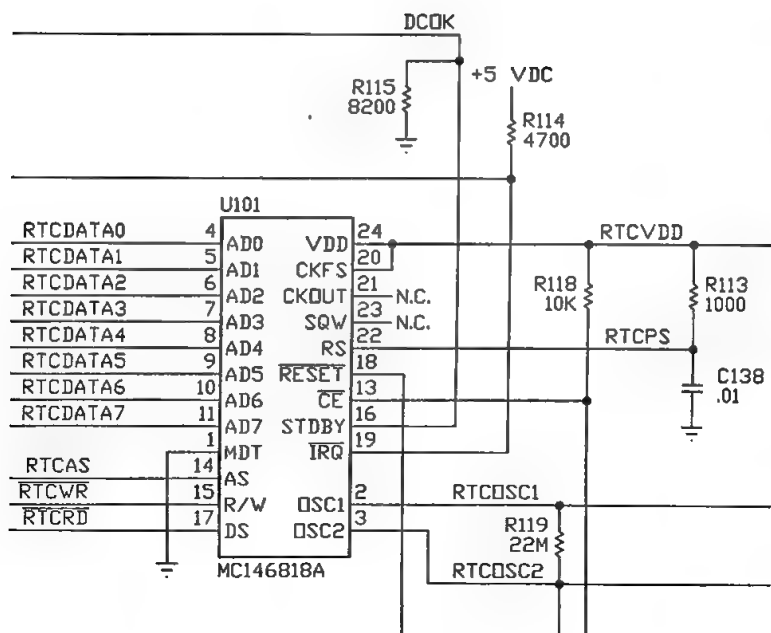


Figure 14-16. Real-Time Clock Pinout

System Control Processor

The system control processor (SCP) is the final major support device used in this computer. The SCP is a single dedicated processor that performs a number of functions in the computer, including:

- Control of keyboard communications
- Control of access to the system slushware
- Monitoring of video output options
- Monitoring of the keyboard inhibit signal
- Support for CPU protected mode operations.

The SCP must accomplish these tasks and operate in a manner that is compatible with normal advanced computer operations. The SCP can also perform special enhanced functions that are not available in other PC-compatible systems. The programmer should be aware that programs that use these special Zenith functions may not operate properly on other PC-compatible computers.

The SCP is an Intel 8042 Universal Peripheral Interface, 8-bit Slave Microcontroller. It contains a specific control program that dictates how the device will respond to any combination of given inputs. Although the programmer cannot access the internal control program, various registers and ports are accessible to determine status conditions related to the functions that the SCP controls.

The Status Register

The status register is an 8-bit, read-only register located at I/O port address 64H. The register contains information concerning the status of SCP operations. Table 14-25 defines the bits for this port.

Table 14-25. SCP Status Register Bit Definitions

BIT	DESCRIPTION
0	Output buffer full. When this bit is set, the output buffer is full. The bit will be reset to 0 when the buffer is read. The output buffer should not be read until this bit is set.
1	Input buffer full. When this bit is set, the buffer contains data that the SCP has not yet read. When the SCP reads the buffer the bit will be reset to 0.
2	System flag. The system monitors this bit during the reset routine. When the bit is clear (0), it indicates that the reset was caused by a normal power up condition. After successful completion of the internal self test, the SCP sets this bit to 1.
3	Command/data. The SCP uses this bit to determine whether the data in the input buffer is a command or data. If the bit is set, the SCP interprets the byte as a command. If the bit is clear, the SCP considers the byte as data.

Support Circuits

Table 14-25 (continued). SCP Status Register Bit Definitions

BIT	DESCRIPTION
4	Inhibit switch. If this bit is set, the keyboard is enabled. This bit is updated whenever data is available in the output buffer.
5	Transmit time-out. The SCP sets this bit if it starts a transmission that it cannot complete properly.
6	Receive time-out. The SCP sets this bit when the keyboard controller fails to complete a transmission within the programmed delay period.
7	Parity error. This bit indicates the parity of the last byte received from the keyboard. A 1 indicates even parity; a 0 indicates odd parity. (The keyboard controller transmits with odd parity)

Buffers

The input and output buffers allow data to move in and out of the SCP. Both of these buffers are 8-bits wide with distinctly different operations.

The input buffer is a write-only register located at I/O port address 60H or 64H. The difference in the addresses indicates the contents of the register. A write to I/O address 60H indicates that the buffer contains data. A write to I/O address 64H indicates that the buffer contains a command for the SCP. Data received at I/O address 60H is normally sent to the keyboard controller unless the SCP is expecting data following a command. Do not write to the input buffer unless bit 1 of the status register is clear.

The output buffer is a read-only register located at I/O address 60H. This buffer contains scan codes from the keyboard controller or data bytes requested by the system. This buffer should only be read when bit 0 of the status register is set.

SCP Commands

Whenever a write to I/O port address 64H occurs, the SCP expects to receive a valid command instruction. Table 14-26 describes the commands that the SCP recognizes.

Table 14-26. SCP Commands

COMMAND	DESCRIPTION
20	Read SCP command byte. The SCP places the current command byte in the output buffer.
60	Write SCP command byte. The next byte received at address 60H is taken as the SCP command byte. Table 14-27 describes the bit definitions for this byte.
AA	SCP Self-test. On receiving this command, the SCP will perform internal diagnostics. If no errors are found, the SCP places 55H in the output buffer.
AB	SCP interface test. On receiving this command, the SCP will perform tests on the keyboard clock and data lines. One of the following results will appear in the output buffer: <ul style="list-style-type: none"> 00 — No errors detected. 01 — Keyboard clock low. 02 — Keyboard clock high. 03 — Keyboard data low. 04 — Keyboard data high.
AC	SCP dump. The SCP will return 16 bytes of data, in scan-code format, to the system. This data consists of SCP memory, the input and output port status, and the SCP status word.
AD	Disable keyboard. This command disables the keyboard. The SCP will set bit 4 in the command byte, forcing the clock line low. Data cannot be sent or received.
AE	Enable keyboard. This command enables the keyboard by clearing bit 4 in the command byte.

Support Circuits

Table 14-26 (continued). SCP Commands

COMMAND	DESCRIPTION
B7	Enable slushware write. This command allows the programmer to access the slushware memory area. This command must be followed with a three-byte instruction sequence to enable writes. The required sequence is 76H, 92H, and 04H. If this sequence is altered, or not supplied, unpredictable results may occur.
B8	Disable slushware writes. This command essentially returns the slushware to its normal write protected state. Like with the previous command, a three-byte command sequence must follow the command. The required sequence is F8H, 4BH, and 69H.
B9	Return SCP version. This command returns a one-byte BCD value to check the device version number. For example, if the device returns the value 22H, then the version number is 2.2.
C0	Read input port. The SCP will read the input port and place the data in the output buffer. This should only be done when the output buffer is empty.
D0	Read output port. The SCP will read the output port and place the data in the output buffer. Like with the read input command make sure the output buffer is empty.
D1	Write output port. The next byte of data received at port address 60H will be placed in the SCP's output port. NOTE: Bit 0 in the SCP output port is connected to system reset. Do not write this bit low.
E0	Read test inputs. On receiving this command, the SCP will read the T0 and T1 inputs and place the data in the output buffer. Bit 0 represents T0 and bit 1 represents T1.

Table 14-26. SCP Commands

COMMAND	DESCRIPTION
---------	-------------

F0 – FF	Pulse output port. The SCP is capable of pulsing bits 0 – 3 low for approximately 6 microseconds. Bits 0 – 3 of this command correspond to the bits of the output port. A bit is pulsed by placing a zero in that bit location.
---------	---

NOTE: The value FE, if used, will cause the SCP to issue a reset command to the 80386 processor.

Table 14-27. SCP Command Byte Description

BIT	DESCRIPTION
-----	-------------

0	Enable output-buffer-full interrupt. If this bit is set, the SCP will generate an interrupt when it places data in the output buffer.
1	Reserved. This bit should always remain a zero
2	System flag. The SCP places the value of this bit in the system flag bit of the status register.
3	Inhibit override. Setting this bit enables normal keyboard operation.
4	Disable keyboard. Placing a one in this bit location disables the keyboard.
5	Zenith mode. Placing a one in this bit causes the SCP to change modes. The SCP will not check parity or convert scan codes while in this mode.
6	PC-Compatible mode. When this bit is set, the SCP will convert scan codes from the keyboard controller into codes used by PC-compatible computers.
7	Reserved. This bit should always remain a zero.

Keyboard Operations

Probably the one task that takes up the majority of time in the SCP is keyboard control. The SCP receives serial keyboard data, performs parity checks, translates scan codes, and presents the data as a byte of information in an output buffer for the system to use. Each time a byte of data is available in the output buffer, the SCP will transmit an interrupt request for system service. The SCP can also send data to the keyboard.

Receiving Keyboard Data

Data received from the keyboard consists of 11 bits in the format illustrated in Figure 14-17. When the transmission is complete, the keyboard controller disables the interface until the system accepts the transmitted byte. If the received byte contains parity errors, the SCP will have the keyboard controller retransmit the invalid byte. The SCP indicates a received parity error to the system by placing the value FFH in the output buffer and setting the parity bit in the status register. The SCP can also time-out a data transmission from the keyboard controller. If a data transmission is not completed within two milliseconds, the SCP will place FFH in the output buffer and set the receive time-out bit in the status register.

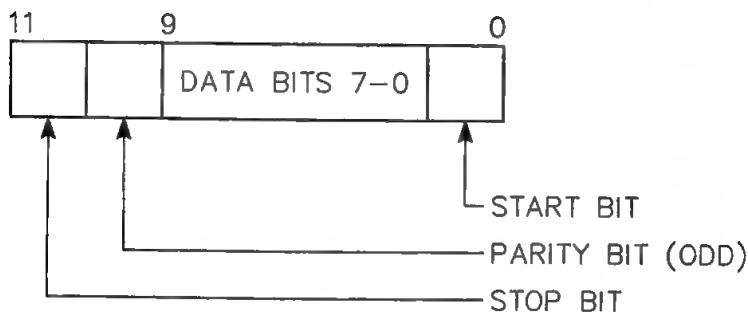


Figure 14-17. Keyboard Data Format

Sending Keyboard Data

Data sent to the keyboard by the SCP uses the same format as that illustrated in Figure 14-17. The data is sent in serial mode with an odd parity bit automatically inserted. The keyboard controller must start clocking the data out of the SCP within 15 milliseconds and complete transmission within 2 milliseconds. If not, the SCP will place an FEH in the output buffer and set the transmit time-out error bit in the status register. The keyboard controller must acknowledge each transmission before another byte can be sent. If a parity error exists in the acknowledgement, the SCP will place a FEH in the output buffer and set both the transmit time-out and parity error bits in the status register. If the keyboard controller does not respond within 25 milliseconds to the SCP transmission, the SCP will place FEH in the output buffer and set the transmit and receive time-out error bits in the status register. By transmitting data to the keyboard controller the SCP controls error re-transmission, repeat and delay rates, and the keyboard LED indicators.

Slushware

Slushware refers to an area of dynamic memory that stores the system ROM code. 128K is always allocated for this purpose. Instructions stored dynamically can be read much faster by the system CPU. For example, BIOS code can be implemented at greater speed when read from slushware instead of from ROM.

When the system initializes, some types of firmware are copied to this area of memory. Zenith products recognize and make use of the slushware area. This can result in significant performance improvements during system operation. In order to take advantage of this performance improvement, the adapter card, such as an EGA video card, must recognize the slushware and make use of it.

The SCP provides a method for the programmer to access the slushware to change or alter the contents of this memory region. Two SCP commands, B7 and B8, referred to in Table 14-26, control this access.

Support Circuits

The slushware area is normally write-protected to prevent accidental alteration of the information stored there. SCP command B7 allows the programmer to perform writes into the slushware area. These memory writes operate in the same manner as writes to other portions of system memory. SCP command B8 will return the slushware to its normal write-protected state.

Both of these commands require a specific three-byte sequence following the command byte. (These sequences are listed in Table 14-26.) Failure to issue these commands in the proper order will cause unpredictable results in the slushware and may halt the computer.

CPU Protected Mode

The SCP assists the programmer in switching the 80386 to protected mode. It manages this by controlling the GATEA20 line in the system hardware.

In older 8088-based computer systems, no address space was available to the processor above address FFFFFH. If a program attempted to address memory above this address, it would wrap around to low system memory. In systems using an 80286 or an 80386, memory can be available above this address space and wraparound is not likely to occur. In order to maintain compatibility with 8088 systems, a method had to be employed to make the hardware resemble an 8088 system. By allowing the SCP to control the GATEA20 line, it can switch between the two hardware configurations.

When the 80386 initializes, it begins operation in real mode. In this mode the processor acts as if it were a very fast 8088. In order to switch to protected mode, the GATEA20 line must be active. This is accomplished by issuing the write output port command (D1) to the SCP. Follow this command with a byte of data having bit 1 set to enable the GATEA20 line. In order for this procedure to work correctly, the program must wait to make sure the line is active. If a program does not wait a sufficient amount of time before attempting to enter protected mode, the results will be unpredictable.

SCP Pinout

Table 14-28 provides a detailed description of the signals associated with the SCP in this computer. Figure 14-18 provides an illustration of the device pinout.

Table 14-28. SCP Pin Functions

PIN NUMBER	SIGNAL NAME	DESCRIPTION
1	T0	Test 0. This signal is used to reset the SCP instruction state and synchronize the internal clock.
2	XTAL 1	Clock input. Not used.
3	XTAL 2	Clock input. Provides the processor clock input signal for device timing purposes.
4	$\overline{\text{RESET}}$	Reset. Resets the processor to a known state and sets the internal program counter to 0.
5	$\overline{\text{SS}}$	Single step. Not used.
6	$\overline{\text{CS}}$	Chip select. An active-low signal on this pin selects the 8042 processor.
7	EA	External access. Not used.
8	$\overline{\text{RD}}$	Read. This signal allows the system processor access to the internal status register or the output buffer.
9	A0	Command/data select. The system processor uses this bit to indicate whether the byte being transferred is a command ($A0 = 1$) or data ($A0 = 0$).
10	$\overline{\text{WR}}$	Write. When this signal is active the system processor can write data or commands to the input buffer.
11	SYNC	Synchronize. Not used.
12 - 19	DB0 - DB7	Data bus. 8-bit, tri-state, bi-directional data bus.
20	GND	Ground.
21	RCAT	80386 reset. Port 2, bit 0. An active-low output signal on this line will reset the system processor.
22	GA20	Gate A20. Port 2, bit 1. The output of this pin will determine the status of the SA20 signal. If this bit is set, the A20 address line will function normally. If this bit is low, the A20 line is disabled.

Support Circuits

Table 14-28 (continued). SCP Pin Functions

PIN NUMBER	SIGNAL NAME	DESCRIPTION
23	—	Port 2, bit 2. Not used.
24	RRWEN	RAM-ROM write enable. Port 2, bit 3. The condition of this bit determines if system is permitted to write data to the slushware. If this bit is set (1), writes are enabled. If the bit is clear (0), no writes are allowed.
25	PROG	Program. Not used.
26	POWER	+5 VDC supply voltage.
27–32	—	Port 1, bits 0–5. Not used.
33	<u>COLOREN</u>	Color select enable. Port 1, bit 6. This bit determines whether a monochrome video adapter (1) or a color video adapter (0) are selected. This signal connects directly to J301 on the system I/O card.
34	KBDINH	Keyboard inhibit. Port 1, Bit 7. This bit enables and disables the operation of the keyboard. If the bit is set (1), keyboard operation is disabled. If the bit is clear (0), the keyboard will operate normally.
35	IRQ1	Interrupt request 1. Port 2, bit 4. The status of this bit reports the condition of this interrupt line. If the bit is set (1), an interrupt request is active. If the bit is clear (0), no interrupt requests are in process.
36	<u>ATKB</u>	Keyboard select. Port 2, bit 5. The setting of this bit determines what type of keyboard is in use. If the bit is set (1), an AT-compatible keyboard is in use. If the bit is clear (0), a PC-compatible keyboard is in use.
37	KBCLK	Keyboard clock. Port 2, bit 6. The clock signal used by the system keyboard is output on this pin.
38	KBDATA	Keyboard data. Port 2, bit 7. Data transmitted to the keyboard is sent via this pin.
39	T1	Timer 1. This line is used as the event timer input.
40	GND	Ground. System ground connection.

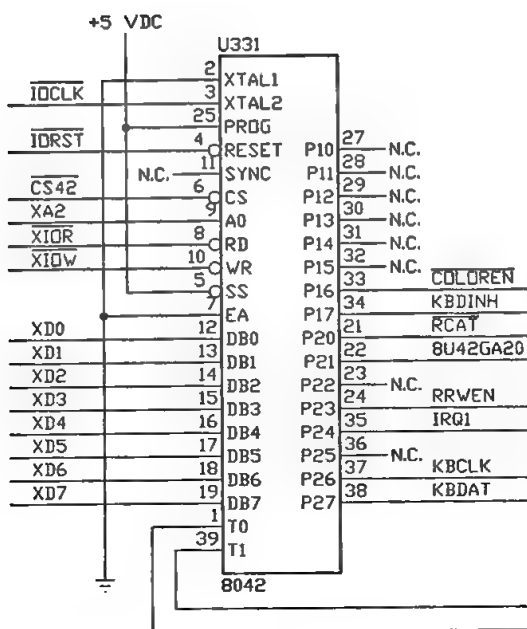


Figure 14-18. SCP Pinout

Special Programming Features

In addition to the programmable functions described throughout this chapter, there are a number of program accessible features that are unique to this system. These features allow the programmer to access special modes or control ports that can enhance the performance of the system. Because these features may not be available on other compatible systems, programs that use them may not perform in the same manner on other systems.

Support Circuits

Fast GATEA20

Speed can be a problem when using the SCP to switch to protected mode. When using this method, a program is forced to wait until the SCP sets the GATEA20 line to an active state. In this computer, an alternative method is available. The hardware contains a special FAST GATEA20 port to allow rapid switches to the protected mode of the processor.

To access the FAST GATEA20 port, the program needs to access I/O port 0EEH. An I/O write to this port will activate the FAST GATEA20 signal. An I/O read to the same port address will disable the signal. This method is recommended over the normal SCP control method because of the increased speed.

When working with the A20 line, both the FAST GATEA20 port and the GATEA20 line must be clear for the A20 line to be disabled. If either of these access methods are active, the A20 address line will remain active.

Fast CPU Reset

The design of this computer allows a program to process a fast reset of the system CPU without disturbing the rest of the system hardware. In order to initiate a fast CPU reset, the program must first enable writes to scratchpad RAM. If the program then performs an I/O read at port 0EFH, the system CPU will be reset.

Scratchpad RAM Enable

Access to the scratchpad RAM is available to the programmer through a special hardware I/O port. If a program performs an I/O write to port 0FBH, writes to the scratchpad RAM will be enabled. An I/O write to port 0F9H will disable writes to the scratchpad RAM.

CAUTION

Be very careful when writing to the scratchpad RAM. Several important system variables are maintained and updated in this area. If a program writes to this area without taking precautions, it is possible to damage the system or halt the computer.

NMI Enable

A special port address exists to enable non-maskable interrupts within the system. If bit 7 at port address 070H is set (1), this feature will be enabled, and will report non-maskable interrupts during system parity errors or if the bus I/O channel check line is active. Clearing the bit at this address will disable this feature.



Chapter 15

Memory

The CPU can address 16 megabytes of memory in its protected-address mode. In real-address mode, it can directly address 1 megabyte of conventional (PC-compatible) memory. The first 640K of this range contains dynamic user memory. The remaining address range (up to 1 megabyte) is reserved for specific controllers (video, hard disk, etc.) and a port through which access to expanded memory (EMS) occurs. Access to memory beyond the CPU's 1 megabyte address range limit can only be accomplished through the 80386 protected address mode or the expanded memory specification (EMS) option.

Additional cards must be added to the system to increase total memory. Each card plugs into the backplane board, which supports three different bus structures: PC-compatible, PC-AT compatible, or Zenith. Electrically, there are two isolated busses: a special high-speed bus and a slower, more compatible bus. The system memory cards available from Zenith Data Systems for this computer support many different configurations. The possible memory allocations for each type of card are detailed in Table 15-1.

Table 15-1. Memory Configuration

	CONVENTIONAL (0-640K)	PROTECTED MODE (1M-16M)	EMS	SLUSHWARE
1-megabyte card	512K	0K	0K	128K
	640K	0K	0K	128K
	512K	0K	384K	128K
	640K	0K	256K	128K
4-megabyte card	512K	3072K	0K	128K
	640K	3072K	0K	128K

Memory

Table 15-1 (continued). Memory Configuration

CONVENTIONAL (0-640K)	PROTECTED MODE (1M-16M)	EMS	SLUSHWARE
512K	1024K	2048K	128K
640K	1024K	2048K	128K

NOTE: The first memory card (card #1) in the system **must** be a 32-bit Zenith memory card.

The memory in this computer system consists of four functionally different sections: conventional system memory (base memory up to 640K and extended memory beyond 1 megabyte), slushware memory, expanded memory (EMS), and cache memory. The following paragraphs give a brief description of each type of memory and the following sections describe them in detail.

Conventional base memory is addressed from 0K - 512K or from 0K - 640K. Configuration switches designate the upper address limit. When base memory is set for 512K, processor access is slowed down between the range of 512K and 640K (PC-AT base memory range). This allows compatibility with controllers that may attempt to use this address space. The processor access is always slowed down between the 640K - 1 megabyte address range because some controllers (video cards, game cards, etc.) use this space exclusively. However, access to slushware memory (which resides at the upper base memory address limit) is not slowed. Extended memory is addressed beyond the 1-megabyte boundary. Access to memory in this region can only occur through the 80386's protected address mode.

Slushware refers to an area of dynamic memory that stores the system ROM code. 128K are always allocated for this purpose. Instructions stored dynamically can be read much faster by the system CPU. For example, BIOS code can be implemented at greater speed when read from slushware instead of ROM.

This computer provides hardware support for expanded memory (EMS). A hardware-fixed window in segment D of the system ROM memory range allows access to EMS memory pages. The CPU addresses segment D and software causes the contents of specific EMS memory locations to be made available. The information is then fetched by the CPU through segment D. Actual use of the EMS memory area requires a special software driver.

An optional system cache card provides 64K of high-speed static RAM between the CPU and the normal system memory. Although the cache is small compared with system memory, it is very fast. It can be thought of as a high-speed buffer that stores data or instructions resulting from the most recent processor operation. A logic sub-system continually updates the information stored in the cache, relative to CPU execution. Because the cache contains current information, the CPU can take advantage of the property of program locality. This property allows significantly decreased memory access time which greatly enhances processor throughput. The cache system is described in more detail later in this chapter.

System Memory

System RAM is organized as specific sized pages according to the type of memory card installed in the system. Two cards are available: a 1-megabyte memory card (model Z-505) and a 4-megabyte memory card (model Z-515). The 1-megabyte card uses four banks of nine 256-kilobit memory chips and the 4-megabyte card uses four banks of nine 1-megabit memory chips. There are eight memory bits associated with each memory row, or 32 memory bits for each card.

NOTE: The memory devices used on the 1- and 4-megabyte memory cards are not interchangeable.

The total number of memory bits multiplied by the internal matrix size of the memory chips determines page size. For example, the 256-kilobit memory chips on the 1-megabyte memory card have a 512-bit \times 512-bit internal matrix. Since there are 32 total memory bits, the page size equals 512 \times 32, or 2K. On the 4-megabyte memory card, the memory chips have a 1024-bit \times 1024-bit internal matrix. The page size of this card is 1024 \times 32, or 4K.

Memory

The memory devices in each page are dynamic enhanced page-mode RAM components. The organization of the memory cells is a two-dimensional matrix which allows very fast access. A specific cell is accessed through row-address and column-address signals. The memory access process takes part in two distinct phases: the row-access phase and the column-access phase. The row-address and column-address signals are independently transferred to the memory device during each phase.

After the row-address is valid on the bus, the row-access phase occurs. When the particular row of memory devices has been selected, the column-access phase is initiated. During the column-access phase any of the memory columns within the enabled row can be read at a very high speed.

There are three types of memory cycles: memory read, memory write, or refresh. Any cycle can be requested by more than one device. Contention, timing, and row and column address selection is governed by the address control logic.

Refresh for a particular address is automatic whenever that location is read. During refresh cycles, memory rows in all pages are read simultaneously through the RAS and CAS signals, automatically refreshing the memory. By multiplexing the address, nine lines are used to address the first 640K of memory.

System base memory is located on the first memory card in the system and contains 512K (or 640K) directly addressable by the 80386 CPU. The address limits are specified through adjustment of the hardware switches described in Chapter 4. The last 128K of the system RAM space stores a copy of the system ROM codes as follows: the Monitor ROM occupies the first 64K, the next 64K may be entirely allocated to optional ROM code or 16K may be reserved for EGA ROM code. The remaining memory, either 256K or 384K depending on the base configuration, can be used as EMS memory.

Additional memory increases the computer's dynamic storage capability. This memory is accessed either through the 80386 protected mode (extended memory) or address space within the reserved memory area beyond the 640K boundary (expanded memory). The

CPU cannot directly address expanded memory. The EMS upgrade allows the CPU to indirectly use the expanded memory. Refer to Chapter 4 for more information on memory configuration options.

Figure 15-1 illustrates the system memory mapping for this computer. The memory map represents the range of base memory, the system ROM code storage area (640K – 1 megabyte, including slushware), extended memory (1 megabyte – 16 megabytes), and expanded (EMS) memory.

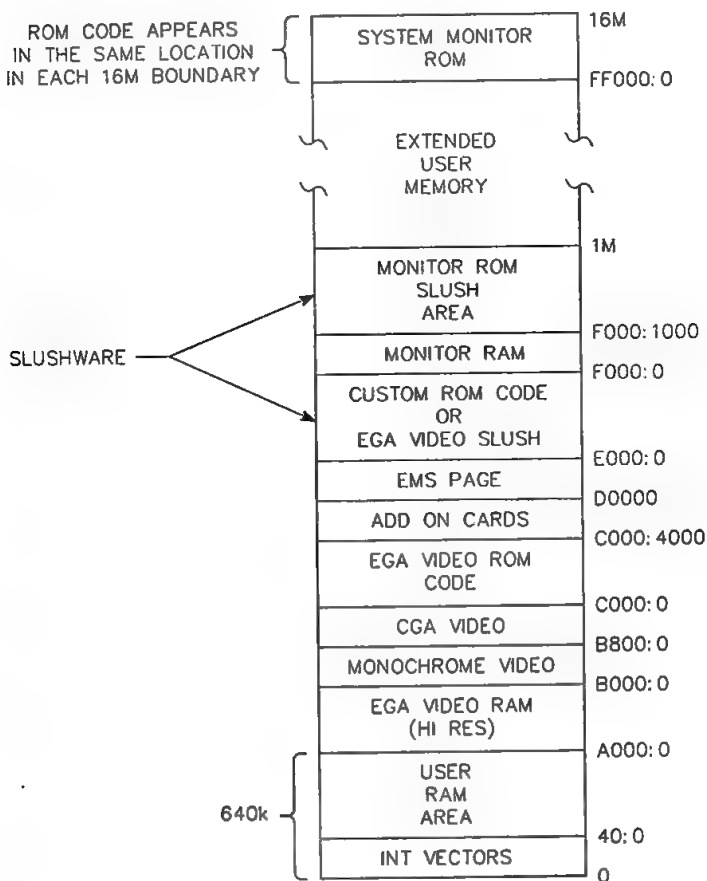


Figure 15-1. Memory Map

System Memory Addressing

The 80386 is capable of generating system memory addresses in several different ways, depending on its operational mode. The simplest mode, real mode, allows the 80386 access to the first megabyte of memory in the same manner as an 8088 microprocessor. In other words, a base address and an offset segment are used to define a specific memory location. In protected mode, or virtual mode, the 80386 calculates memory addresses differently. Additional factors that contribute to the memory address mechanism involve the internal paging mode of the 80386. The following discussion describes the memory addressing methods used by the 80386.

User Memory

Most user memory programming can be performed directly from assembly or machine language programs and includes the following:

- Reading and writing data to specific memory locations
- Allocating or reserving specific locations in memory for specific purposes, such as for software interrupt routines or storing user-defined character fonts
- Storing character strings and numeric values that are widely used by a number of applications
- Rerouting interrupts to user-defined routines
- CPU stack operations
- Moving memory contents from one area to another
- Using the contents of RAM to control video graphics.

Memory Address Format

The 80386 uses a much more involved method of address computation than the earlier 8088 microprocessor but similar to the method employed for the 80286 microprocessor. Because of the computation methods used, programming situations may arise where it is extremely difficult to identify the actual memory being addressed. Since special translation units within the 80386 control this address calculation, this should not be a critical programming issue.

The earlier 8088 uses a 2-part number to designate specific memory locations. The hexadecimal number consists of a 4-digit number to identify the segment address and a 4-digit number to identify the offset address within that segment. The format for this value is XXXX:YYYY.

The first four digits actually represent a 5-digit hexadecimal memory address; the system performs an imaginary shift left (multiply by 16) on the value to arrive at the RAM bank and row to select. The second value selects the RAM column from the selected bank. For example, the value 3F3F:5B11 represents memory address 44F01H. 3F3FH shifted left equals 3F3F0H. 5B11H is the offset added to the segment address. The result of adding 3F3F0H and 5B11H is 44F01H.

In the 80386 the concept of the segment and offset have remained but the computation methods have changed. The 80386 refers to three distinct address spaces: logical, linear, and physical. Figure 15-2 illustrates the address translation process. Refer to this figure for the following discussion.

The logical address (referred to as the virtual address) consists of a selector and an offset. The selector is a value contained within a segment register. The offset (referred to as the effective address) is composed of a combination of three different values: the base, the index, and the displacement. How these values are combined depends on which one of 11 different addressing modes is in use. The resulting virtual space is the area the programmer will most often encounter and use.

Memory

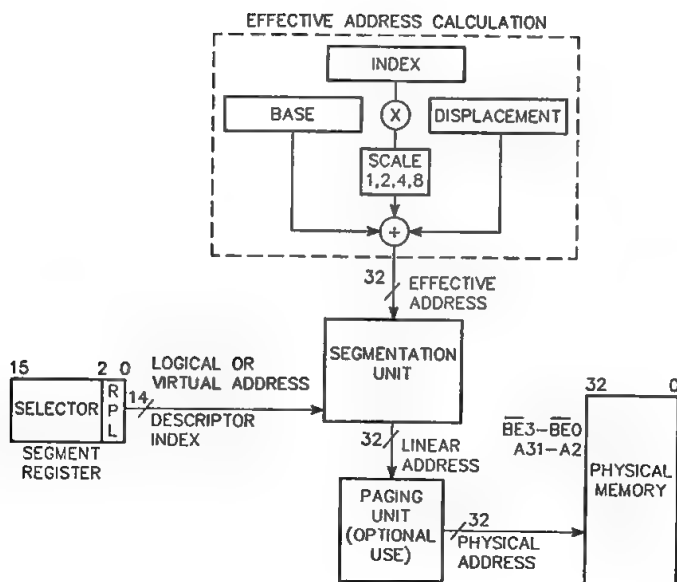


Figure 15-2. 80386 Address Translation Process

A special segmentation unit within the 80386 processes the logical address information to obtain the linear address. The control for this process is the processor mode. The 80386 can operate in real mode or protected mode and each mode has a different process to determine the linear address.

In real mode, the segmentation unit shifts the selector value left four bits and adds the result to the offset to form the linear address. If the internal paging unit is not enabled, the linear address becomes the physical address in memory. (This method duplicates exactly the method used in the 8088 processor.) If the internal paging unit is enabled, it performs an additional translation on the linear address to determine the physical address.

In protected mode, each selector value is associated with its own linear base address. These values are stored in special descriptor tables within the 80386. The selector's linear base address is added to the offset to form the final linear address. The internal paging unit operates the same as in real mode on this linear address.

Address Decoding

The address control logic decodes signals that control specific operations such as memory access and refresh. Also, some signals supplied by active circuits on other cards in the system are transferred across the backplane board to be decoded by the memory card. Programmable logic arrays provide fundamental decoding in conjunction with an array of multiplexers. The logic arrays are programmed to interpret specific pulse definitions and respond by asserting a predetermined signal at a predetermined output.

The status of the hardware switch segments also are applied to the inputs of the logic arrays. The settings of the switches represent two states that the logic array interprets and reacts to. Some outputs of the logic arrays are gated by discrete logic to produce the required signals used during memory read cycles, memory write cycles, and refresh cycles. Other signals control which memory bank is selected, and whether a memory cycle or refresh cycle occurs.

The array of multiplexers generate the MA0 – MA13 (memory address bits 0 – 13) address lines from signals picked off of their inputs. A series of jumpers are placed to synchronize multiplexing (and therefore memory addressing) against refresh cycles. Refer to Chapter 4 for a complete description of these jumper settings.

Data Bus Interface

Data interface between memory and the CPU is handled through four bidirectional data transceivers. Each transceiver is capable of transferring 8 bits of data. During a given memory access (read or write), one 32-bit segment, one 16-bit segment, or one 8-bit segment of data can be transferred through these transceivers.

Directional control for the transceivers is provided through combinations of the memory bank enable signals, memory read signals, and memory write signals. Therefore, 8 bits of data can be transferred through any one transceiver relative to its selected memory bank.

Memory

Memory Data Bus — The memory data bus is 32 bits wide and is interfaced to the system data bus through four separate transceivers. Each bit of the memory data bus (MD0 – MD31) corresponds to each bit of the system data bus (D0 – D31). Each group of eight data lines is assigned to a specific bank of memory as indicated in Table 15-2.

Table 15-2. Memory Bank Assignments

DATA LINES	MEMORY BANK
D0 – D7	Bank 0
D8 – D15	Bank 1
D16 – D23	Bank 2
D24 – D31	Bank 3

The memory address lines (MA0 – MA10) enable one specific memory location within the active memory bank (each bank is a row of memory chips). On the base memory card, only MA0 – MA8 are active. If additional memory cards are present and have 1-megabit RAM devices installed, MA0 – MA9 are active. If 256-kilobit devices are in use, only memory lines MA0 – MA8 will be active. The additional lines are necessary for the system to address to the 1-megabyte boundary.

Each memory data line (MD0 – MD7) is tied to the bidirectional data input/output pin of one memory device in each memory bank. Data is either read from or written to the same memory location in each device during a memory cycle. A valid address on the MA1 – MA13 (memory address) bus and the state of the WE (write enable) signal define the operation (high to read, low to write). Figure 15-3 illustrates a portion of the memory hardware configuration.

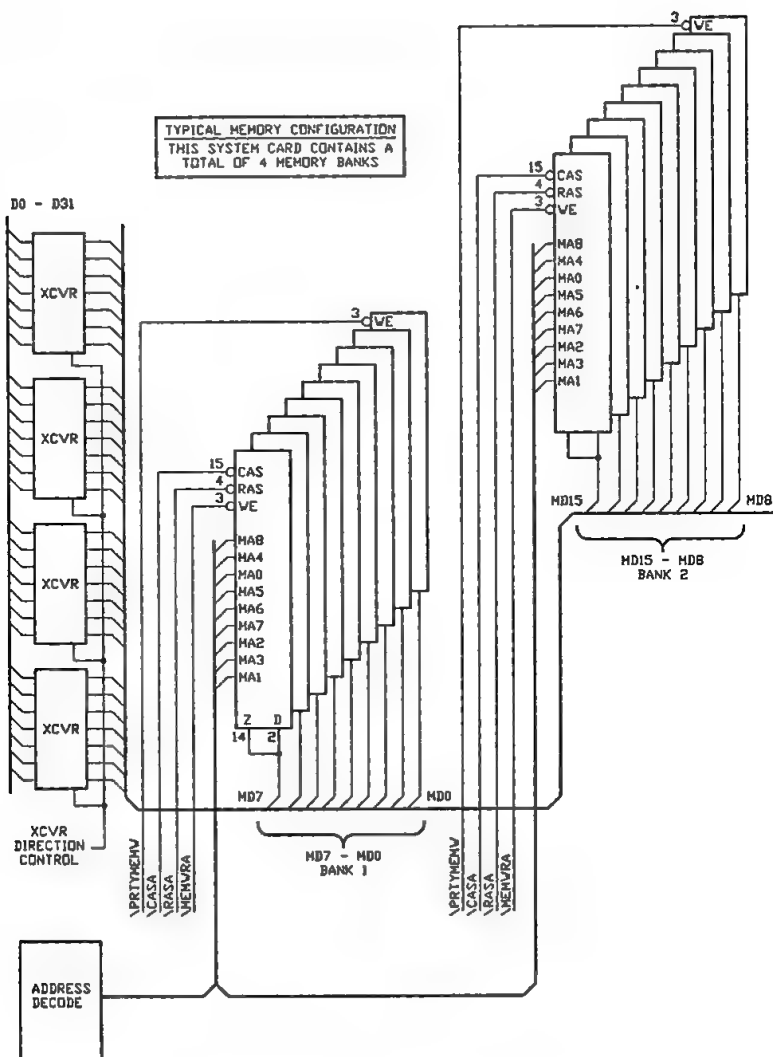


Figure 15-3. Memory Data Bus

Memory

Paging Controller

The paging controller is a special circuit that dynamically controls the speed at which memory access occurs. Basically, the paging controller determines whether the memory page to be accessed is the current page or a different page. It does this by comparing the current memory address to the previous address. If the current address is within the same page as the last cycle, the operation will occur without wait states to enhance system throughput. If the address specifies a different page, the cycle is dynamically slowed, using wait states, to a rate comparable to conventional access methods.

Refresh (RAS/CAS)

The dynamic RAM devices that make up system memory must receive refresh pulses at least every four milliseconds to prevent data from deteriorating beyond the point of validity. Most computers refresh single rows of memory devices. To do so, the CPU must relinquish the bus to the refresh control logic during each cycle. This can cause the efficiency of the processor to be considerably degraded.

In this computer, an optional refresh bursting method is incorporated to allow multiple refresh operations to occur as one cycle. In other words, rather than refreshing one row of memory during a refresh cycle, this technique allows up to four rows of memory to be refreshed during the same cycle. This results in improvements in system throughput because the processor is not giving up the bus as often.

Bus arbitration is maintained through logic arrays in the address decoding circuits. This control prevents refresh cycles from occurring during some other memory cycle. If refresh did occur at the wrong time, it could corrupt the state of data or instructions currently in memory and the system would likely be halted.

Parity

Parity is a method of continually testing the integrity of data in memory. These tests check only the memory devices and will not detect errors due to other hardware or to system bus contention. The parity testing in this computer is similar to that used in other Zenith Data Systems advanced personal computers. This computer system supplies odd parity status for 8-bit byte accesses to memory, 16-bit word accesses to memory, and 32-bit double-word accesses to memory.

In general, for every 8-bit data byte written to each system memory bank, one bit will be written to the parity memory device. The parity memory device is the same type used for system memory. The status of the parity bit is compared with its associated data byte during each memory read cycle, to determine if an error has occurred.

Generator — The parity generator circuits are discrete devices that supply an odd parity bit in this computer. Each memory bank's parity generator monitors the value of all data bytes written to that bank. If a data byte contains an odd number of high bits (logic one), the parity generator writes a logic one to its associated parity memory device.

Checker — Parity checking involves the same set of circuits that generate parity. Checking occurs in synchronous fashion during memory read cycles. When a data segment is read from a memory bank, it is placed on the MD0 – MD31 bus and applied to the parity circuits. The parity circuits evaluate the data segment expecting to detect the same odd number of high data bits that were written to memory.

If a memory device were defective, the status of a data bit previously set to a logic one could deteriorate to a logic zero state. This defect will cause the total number of data bits previously set to one to equal an even number. The parity checker senses this condition and drives its output low.

Memory

A logic array works in conjunction with the parity circuits to alert the system when such errors are detected in memory. The parity logic array senses the change in the parity generator's output and reacts by asserting the IOCHK signal to the system CPU. The IOCHK signal is latched, representing an error condition. An I/O write operation (consisting of any data value) to port 0F7H is required to clear the error condition.

Because the IOCHK signal is latched, detailed analysis of the error can be made through error recovery software. The memory defect can be pinpointed to a specific memory bank on a specific memory card in the system. Because each memory card responds to a different port address, the system can periodically address each port and obtain error status for each card. Table 15-3 describes this error reporting scheme.

Table 15-3. Memory Board Error Reporting Scheme

I/O PORT	MEMORY CARD
F4H	Card 0
F5H	Card 1
F6H	Card 2
F7H	Card 3

NOTE: The memory card number is determined by the configuration switch settings described in Chapter 4.

After a specific memory card has been isolated, each memory bank on that card can be checked for an error condition. Table 15-4 indicates how error reporting is handled for each memory bank of a selected card.

Table 15-4. Memory Bank Error Reporting Scheme

DATA LINE	MEMORY BANK
D0	Bank 0 (MD24 – MD31)
D1	Bank 1 (MD16 – MD23)
D2	Bank 2 (MD8 – MD15)
D3	Bank 3 (MD0 – MD7)

If an error condition exists, it is represented by the latched status of the IOCHK signal. To clear the error and reset the latch, an I/O write operation to port F7H is required. The value of the data written to the port is not critical and the error status of all memory banks on all Zenith boards will be cleared.

Parity Hardware Test — The $\overline{\text{DISPG}}$ (disable parity generation) signal line on the system bus can test the parity generating/checking hardware. When the $\overline{\text{DISPG}}$ signal is active, it masks write operations to the parity circuits. Random data can then be written to the system memory. If successive memory read cycles are initiated, the parity hardware should detect a sequence of errors. The integrity of the parity checking circuits can then be determined.

Memory Bank Configuration

There are four memory banks on each memory card that can be used with this computer. The basic system is populated with 1 megabyte or 4 megabytes of memory, depending on the system base configuration. The first 512K (or 640K) is dedicated as system memory. The memory devices are 256 kilobit integrated circuits (on the 1-megabyte card) and the memory banks are nine devices wide (eight devices for data or instructions and one for parity).

Up to four memory expansion cards can be installed in this computer. Also, each memory bank can be populated with 1-megabit memory devices. Data transfer operations involving these memory banks are the same as for the base memory card. Access to memory locations above the 1-megabyte boundary can occur in one of several ways. The 80386 can access this memory in protected mode, a portion or all of the memory can be accessed as EMS (expanded memory specification) memory using a software device driver, or the memory can be accessed as a virtual disk using the appropriate device driver.

Memory

Slushware

The term "slushware" refers to a technique that combines the advantages of software and firmware for increased flexibility and speed. At system initialization, the Monitor ROM BIOS routines are downloaded into a reserved area of system memory. Since the BIOS resides in RAM during normal operations, the routines execute much faster. To prevent the BIOS instructions from being corrupted by a user program, the slushware memory is write-protected.

This computer also supports additional user-specified system ROM, which can also take advantage of the slushware concept. A socket on the CPU board accepts an optional system ROM device (frequently used by OEM groups) that gives an additional 64K for unique code applications. The code can also be downloaded to slushware RAM from diskette.

If the computer contains an enhanced graphics adapter, a special slushware routine places the EGA ROM code in slushware RAM. Special techniques modify the EGA code to maintain compatibility while substantially increasing the speed of the video display. When slushware memory contains the Monitor, system, and EGA ROM codes, the first 64K is reserved for the Monitor program, the next 48K for system ROM code, and the remaining 16K for EGA code.

Another advantage in slushware memory is that instructions stored there will be placed in the optional cache memory (if the cache controller card is used) whenever the CPU addresses slushware. This process results in greater speed and system throughput because information is local to the CPU.

EMS Memory

This 80386-based computer directly addresses the first megabyte of contiguous memory in its environment. This system memory is conventional in PC and PC-AT computer products. Memory from 640K to the 1-megabyte address boundary has portions reserved for system ROM code, video controller ROM code, hard disk controller ROM code, and other functions. Memory from 1 megabyte to the 4-gigabyte address boundary is termed extended memory and is accessible only through the protected mode of the 80386. In contrast, expanded memory (EMS) is page-mapped memory accessed through a complicated bank switching scheme in the 80386 real address mode.

Expanded memory is only useful to programs that support it. In other words, the program must instruct the hardware to implement the specific bank switching. The Lotus/Intel/Microsoft Expanded Memory Specification (LIM EMS) attempts to standardize the bank switching scheme. The LIM EMS contains numerous examples that an experienced assembly language programmer can interpret. In this way, different programs can be written to make use of expanded memory in a consistent manner. The specification is available free of charge from Intel Corporation and may be obtained by calling 1-800-538-3373.

This following section briefly outlines the approach used in the LIM EMS to access expanded memory. As mentioned earlier, expanded memory is page-mapped. This means that, a small area of memory acts as a window through which a larger area of memory resides. This window is 64K wide and is composed of four 16K blocks of memory referred to as page frames. In this computer these page frames must be contiguous and exist in segment D (address range D0000H – DFFFFH) of the reserved system ROM space (between 640K and 1 megabyte). In other words, this computer is hardware configured to support expanded memory through segment D.

Memory

Some computer systems allocate some of the system ROM space to optional hardware equipment (EGA video, hard disk, etc.). In these systems, it may be difficult to find a contiguous 64K section. However, the first page frame (the base-address of the 64K window) must fall on a 16K boundary within the system ROM space as illustrated in Figure 15-4.

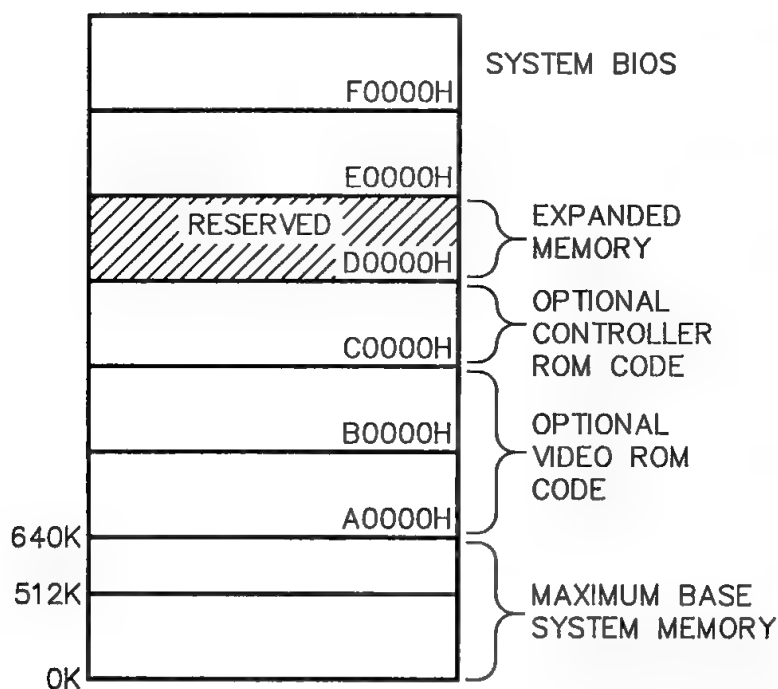


Figure 15-4. Conventional Memory

The EMS specification supports up to four memory cards, each populated a maximum of with 2 megabytes of memory. Even though this computer supports a maximum of four 4-megabyte memory cards, only 8 megabytes of expanded memory can be used, according to the LIM EMS. The memory on each card is organized into 128 individual pages of expanded memory and each page frame is 16K wide. Because the CPU does not have direct addressing capabilities beyond 1 megabyte, it uses the page frames to access expanded memory (recall that the page frames reside within the 640K to 1-megabyte memory range). Whenever the CPU addresses the memory range within system ROM space that is associated with a page frame, it accesses expanded memory. Since there are four contiguous page frames, 64K of expanded memory can be accessed without a page-swap occurring. Refer to Figure 15-5.

NOTE: The page pointing arrangement illustrated in Figure 15-5 is not the only pointing configuration possible with EMS memory. Many different arrangements are possible, depending on the needs of the software being used.

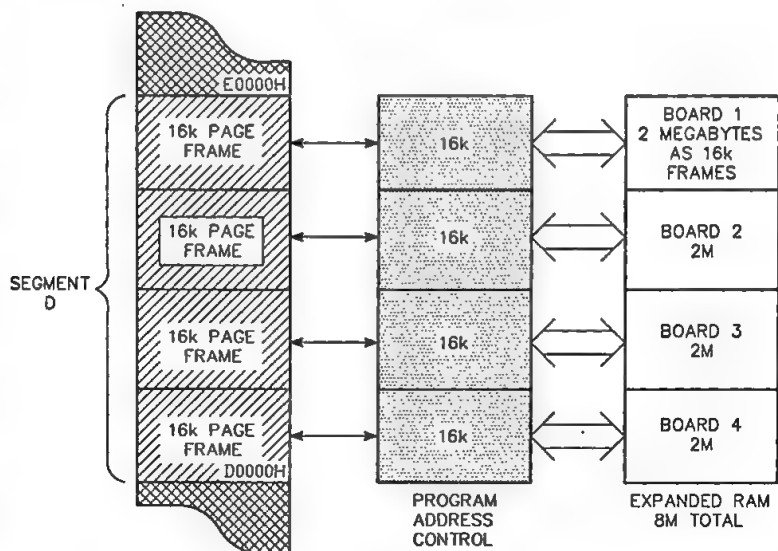


Figure 15-5. Expanded Memory

Memory

As the CPU addresses the memory range of one of the page frames, the EMS memory page pointer will remain set to that frame. If the page frame changes or the page is turned on or off, then an 8-bit value is written to an I/O port to select the correct frame. The value placed in the port moves a specified 16K expanded memory page to the page frame. The information is thereby made local to the CPU.

Bits 0-6 of the page pointer byte define the particular page, while bit 7 is the page enable/disable bit. When bit 7 is set (logic high), the page is allowed to appear in the memory space; when bit 7 is clear (logic low), the page does not appear in the memory space. Refer to Table 15-5 for the EMS I/O addresses for this computer.

Table 15-5. EMS I/O Addresses

MEMORY CARD	PAGE 0	PAGE 1	PAGE 2	PAGE 3
Card 0	0258H	4258H	8258H	C258H
Card 1	0268H	4268H	8268H	C268H
Card 2	0208H	4208H	8208H	C208H
Card 3	0218H	4218H	8218H	C218H

NOTE: Card numbers are switch selectable. Only one page per card may be enabled at one time.

EMS address decoding up to the 1-megabyte boundary occurs as normal. The logic array in that circuit asserts the required control signals and the multiplexer array provides the memory address information. For more information refer to the "Address Decoding" section under "System Memory" earlier in this chapter. Access to specific memory pages is a function of the program that is executing at the time.

Memory refresh cycles occur for expanded memory in the same way as for system memory. Memory refresh is handled during the appropriate processor cycle and addressing is manipulated so the refresh cycle includes expanded memory. For more information refer to the "Refresh" section under "System Memory" earlier in this chapter.

Data is transferred between the CPU and EMS memory in the same way that data is interfaced to system memory. The same data transceivers and control signals are active. For more information refer to the "Data Bus Interface" section under "System Memory" earlier in this chapter. The parity circuits used for system memory are also used to check the integrity of transferred EMS data. The parity bit is computed during the write cycle and checked during the read cycle, as described in the "Parity" section under "System Memory".

Cache Memory

Cache memory is a small amount of very high speed RAM between main memory and the CPU. This memory region is transparent to the software and hardware environment of the computer system. The purpose of the cache is to monitor and retain a copy of the most recent program driven CPU operation involving main memory. When data and instructions are read from the cache instead of from main memory, they are provided to the CPU immediately. The CPU will initiate a normal memory access but the main memory controller will not because of the available cache data. Main memory access cycle time is generally slow and often forces the CPU to wait for data or instructions. A cache eliminates the need for a designated memory access cycle; the cache access time is a function of the speed of the cache memory devices. If the CPU must initiate a memory access cycle to main memory, access time is a function of the speed of the main memory devices. Refer to Figure 15-6.

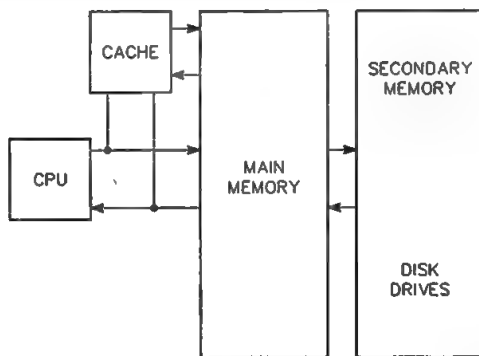


Figure 15-6. Cache Memory Block Diagram

Memory

Cache memory operates on the principle of program locality. In other words, many programs have the tendency to instruct the CPU to access the same area of memory in a repetitive fashion. After the first access, information from that segment of main memory can be duplicated in the cache. Then, when the CPU is instructed to access the same segment of main memory it will retrieve the information from the cache instead of initiating an access cycle to main memory.

There are three components to the property of program locality: temporal, spatial, and sequential. Temporal locality is the tendency of a program to access (in the near future) a memory location that has recently been accessed. An example of temporal locality is when a program contains an instruction loop. Spatial locality is the tendency of a program to access a memory location in the neighborhood of the last access. Sequential locality is the tendency of a program to access the memory location immediately succeeding the one last accessed. The effect of program locality is that only a small amount of cache memory is necessary to be effective.

A logic subsystem controls and updates information in the cache continually. The CPU never has to devote specific cycles to this process. The efficiency of the cache is determined by how often the cache supplies the CPU with information versus the number of accesses made to main memory. The fewer times main memory requires access, the higher the efficiency of the cache. The following sections explain how the cache logic subsystem implements memory read and memory write operations involving cache memory and main memory.

NOTE: There is no programmability in the cache subsystem. It is entirely transparent to the main computer system.

Main Memory to Cache Memory Mapping

Cache memory contains 64K of high speed static RAM. The 32-bit data path between the CPU and main memory passes directly to the cache memory. Therefore, a 64K cache with a 32-bit data path includes 16K blocks of memory, each block consists of four bytes. A direct mapping function determines which area of main memory addressed by the CPU, transfers into cache memory. For example, whenever four bytes of main memory is addressed, it can be mapped into the first 16K block of the cache.

Memory Read

There are two types of read operations involving cache memory: cache hits and cache misses. When a cache hit occurs, the information contained in the program-specified main memory location is already present in cache memory. If a cache miss occurs, the information is not in the cache and the cache must initiate a designated memory access cycle to main memory. The program-specified memory location is then addressed and its contents are transferred to the CPU and stored in the cache.

For each memory read, the cache must be checked to determine if it contains the program-specified information (to determine if the read is a hit or a miss). A comparison is made between the address generated to fetch data from main memory and the memory address range currently contained in the cache. Each address has a block portion (low-order bits) and a tag portion (high-order bits) associated with it. The tag portion determines if the address range that holds required data or instructions is in the cache, and the block portion selects the particular memory location (either in the cache or main memory) specified by the program.

Memory

When an address is generated, the block portion (low-order bits CA2 – CA15) is applied to a fast memory device called a tag RAM. The output of the tag RAM is fed to a comparator circuit. The tag portion of the address (high-order bits CA16 – CA23) is also applied to the comparator circuit. If the output of the tag RAM is equal to the tag portion of the address, then the address range containing the required data and instructions is in the cache. The information is transferred to the CPU through the high-speed data RAM without wait states. Refer to Figure 15-7.

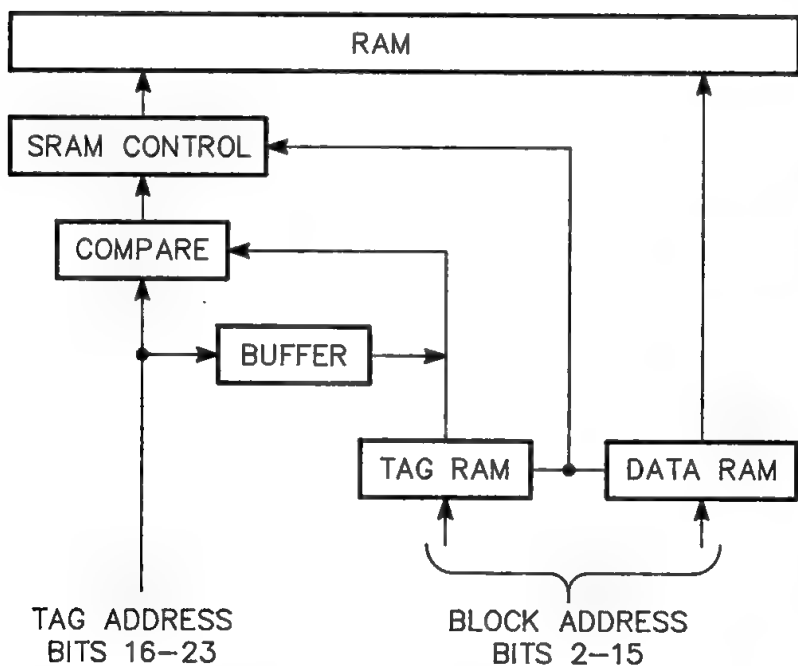


Figure 15-7. Cache/Main Memory Arbitration Logic

Note that the comparison to determine a cache hit or a cache miss takes place in the processor cycle immediately preceding a main memory access cycle. Therefore, if a cache hit occurs, a main memory access cycle never starts. If a cache miss occurs, the cache does not contain the required information and a main memory access cycle will occur within the same number of processor cycles.

When a cache miss occurs, the high-order address that specifies the main memory location (in conjunction with the low-order address) is written into the tag RAMs. As the processor receives the required information from the main memory access, the cache is automatically updated to include that same information. The algorithm that implements this operation is depicted in the flowchart in Figure 15-8.

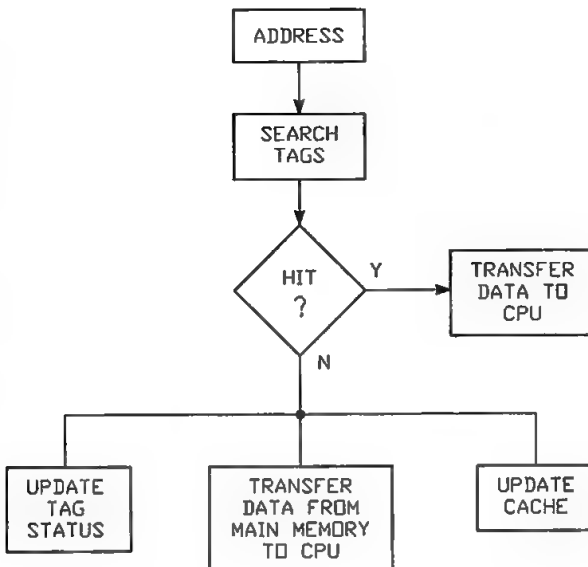


Figure 15-8. Cache Read Miss Logic

Memory

Memory Write

During write operation, information must be updated in the cache (if it contains the address range that the write must occur in) and in main memory. The address generated by the CPU is compared in the same way as for memory read operations. If the comparison determines a cache write hit has occurred (the tag portion of the address agrees with the block portion of the address), the memory location to be updated is in the cache. The cache and main memory will be updated at the slower speed of the main memory.

If a cache write miss occurs (the tag and block portions of the address do not equal), the cache does not contain the memory location to be updated. Therefore, only main memory is updated. Note that main memory is updated in every write cycle. The cache is only updated when it contains the same memory location as main memory.

Dynamic Memory Control

In this computer, control operations for memory accesses are normally handled by circuits on the CPU card. However, if a cache controller card is installed, it assumes control for memory access operations. The cache controller card actually contains a main memory controller circuit and a cache controller circuit. These circuits are very tightly coupled and work together to adjust memory accesses to the speed of main memory, when required. Memory cycles in this computer are dynamic and can contain different numbers of wait states.

Chapter 16

Mass Storage

The term "mass storage" applies to all forms of off-line storage including magnetic disk, magnetic tape, punched tape, and punched cards. This computer supports several types of mass storage devices including:

- 5.25-inch, 1.2M byte floppy disk drives
- 5.25-inch, 360K byte floppy disk drives
- 3.5-inch, 720K byte floppy disk drives
- 3.5-inch, 1.4M byte floppy disk drives
- 5.25-inch, 10M byte removable cartridge hard disk drives
- 3.5-inch, 20M byte fixed hard disk drives
- 5.25-inch, 20M, 40M, and 80M byte fixed hard disk drives.

This chapter discusses the programmable elements of the mass storage system in this computer.

Supported Drives

A maximum of two floppy disk drives may be installed in this computer. The operating system and the devices used in this computer will support dual-speed, high-density 5.25-inch floppy disk drives, high-density 3.5-inch floppy disk drives, and standard-density 5.25-inch floppy disk drives. However, 8-inch external floppy disk drives are not supported. Do not attach 8-inch drives or program the computer or its operating system for 8-inch operation. Contact your service representative for information about the floppy disk drive which best suits your needs.

A maximum of two half-height or full-height hard disk drives may be installed. Both 5.25-inch and 3.5-inch drives are supported. The 5.25-inch drives are 20M, 40M, or 80M fixed disk drives or 10M removable cartridge drives. Some 3.5-inch, 20M fixed disk drives (mounted in a 5.25-inch chassis) are also used.

Drive Selection

You may select the drive type using either hardware or firmware. Jumper blocks W5 and W6 determine which mode is in use. In hardware drive select mode, the drive number in the drive/head register is driven directly onto the drive interface. In firmware drive select mode, the drive number is intercepted by the controller's firmware and the firmware-selected drive number is driven onto the interface. The computer is factory-set for hardware drive select mode.

To use hardware drive select mode:

1. Install a jumper on W5.
2. Install jumpers across pins 2 and 3 and across pins 5 and 6 on W6.

To use firmware drive select mode:

1. Leave W5 open.
2. Install jumpers across pins 1 and 2 and across pins 4 and 5 on W6.

Error Detection and Invalid Commands

Four error correction code (ECC) bytes are used for error detection and correction in the data fields. This permits correction of five bytes in the data fields. A cyclical redundancy check (CRC) byte is used for the ID field. Before calculating the checksums, all shift registers are preset to FH. All checksums begin with the respective address marks. The ECC polynomial is $X^{32} + X^{28} + X^{26} + X^{19} + X^{17} + X^{10} + X^6 + X^2 + 1$. The CRC polynomial is $X^{16} + X^{12} + X^5 + 1$.

If the floppy disk controller receives an invalid command, it is interpreted as a NOP (no operation) command. The controller goes into a standby or no operation state until a new command is issued. If the hard disk controller receives an invalid command, it issues an aborted command error. The error register must be read before the next command will be accepted by the controller.

Floppy Disk Control

The 765 floppy disk controller (FDC) used in most PC-compatible products also provides floppy disk drive control in this computer. This system supports up to two disk drives through a daisy-chained flat cable. The controller makes record data transfers through the I/O bus and DMA circuitry and supports four data transfer rates. Write protect features are recognized and interrupts are used to indicate completion of an operation. The FDC supports FM and MFM disk encoding formats and can be operated in DMA and non-DMA modes.

DMA and Non-DMA Modes

This computer is factory-set to operate in DMA mode. In DMA mode, the processor loads a command into the FDC and the data transfer is carried out by the DMA and FDC controllers. In most cases, a processor interrupt is generated upon the completion of the data transfer.

In non-DMA mode, the FDC generates a processor interrupt for each data byte transferred. The processor must then load a command into the FDC, allowing the single byte to be transferred. To restrict operation to non-DMA mode only, the $\overline{\text{DACK}}$ line (pin 15 of the $\mu\text{pd}765\text{A}$) can be tied to Vcc . The specify command, described in the "Programming Floppy Disk Operations" section of this chapter, can also be used to operate in non-DMA mode.

Mass Storage

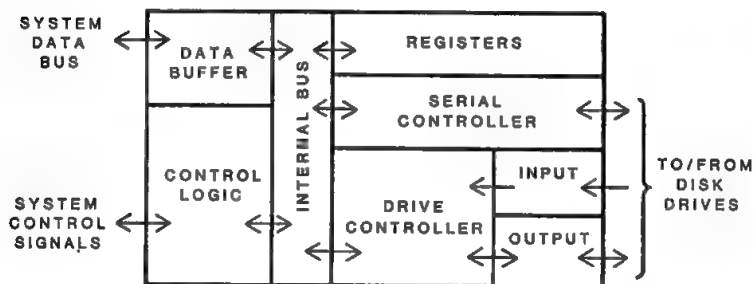


Figure 16-1. 765 Disk Controller Block Diagram

Master Control Logic

The master control logic oversees all operations within the FDC. The control signals it generates handle the data buffer, internal registers, serial controller, and the drive controller through the internal bus. It also handles interrupts, DMA transfers, request and acknowledge signals, and read and write operations.

Serial Controller

An internal data buffer transfers data between the internal bus and the system data bus. The internal bus passes information between the data buffer and the serial controller. The serial controller then reads or writes data from the disk and the registers. The serial controller sends data to the disk drives, converting the 8-bit parallel-format data into a serial form. The controller also converts serial-format data from the drive into an 8-bit parallel form. Using timing control provided by the write clock signal, the controller makes sure that the data is correctly timed during all transfer operations. The serial controller also generates the synchronizing signals used by the data separator for timing.

Drive Controller

The drive controller monitors the activities of the input and output channels of the floppy disk controller. The input channel monitors and receives drive ready, write-protect, index, and track 0 signals. These signals go to the drive controller, which responds by sending signals to either the output channel or the master control logic of the floppy disk controller. The output channel generates seek, head load, head select, direction, and step signals for the drive. It also controls the recording mode (FM or MFM) used by the drive.

FDC Control Registers

Five registers control and provide status reports for floppy disk drive operations. The digital output register and floppy control register are write-only registers that provide command and operating parameter information to the FDC. The main status register is a read-only register that provides drive, FDC, and command status information. The floppy data register is a read/write register used to transfer data, command, and detailed status information between the FDC and the CPU. The digital input register is a read-only register primarily used for hard disk operations. Bit 7 of the digital input register reflects the state of the floppy disk changed signal. Table 16-1 lists the primary and secondary port addresses for these registers. To use the secondary port addresses, install a jumper on W2 and W3.

Table 16-1. Floppy Drive Control Register Port Addresses

PRIMARY PORT ADDRESS	SECONDARY PORT ADDRESS	READ OPERATION REGISTER	WRITE OPERATION REGISTER
3F2H	372H	Not used	Digital output
3F4H	374H	Main status	Main status
3F5H	375H	Floppy data	Floppy data
3F7H	377H	Digital input	Floppy control

Mass Storage

Digital Output Register (Port 3F2H)

The digital output register controls the drive motors and drive selection, enables the disk interrupt (IRQ6) and DMA data request line (DRQ2), and resets the drives. Port address 3F2H is the PC-compatible address used for writing to this register. Table 16-2 describes the function of each bit in this register.

Table 16-2. Digital Output Register

BITS	FUNCTION	DESCRIPTION
0-1	Drive select bits 0 – 1	These bits select which drive is to be enabled. Refer to Table 16-3.
2	$\overline{\text{765RESET}}$	This bit, when cleared, resets the floppy disk control circuits and clears all internal registers.
3	Enable IRQ6 and DRQ2	This bit, when high, enables the IRQ6 (hardware floppy disk interrupt) and the DRQ2 (DMA request) lines.
4	Motor on 0	This bit asserts the $\overline{\text{MTON0}}$ and FDD ON signals to turn on the motor in drive A.
5	Motor on 1	This bit asserts the $\overline{\text{MTON1}}$ and FDD ON signals to turn on the motor in drive B.
6	Motor on 2	This bit asserts the $\overline{\text{MTON2}}$ and FDD ON signals to turn on the motor in a third floppy drive.
7	Motor on 3	This bit asserts the $\overline{\text{MTON3}}$ and FDD ON signals to turn on the motor in a fourth floppy drive.

Table 16-3 defines the drive select bits (bit 0 and bit 1) of the digital output register. The appropriate motor on bit (bits 4 through 7) must also be set in order to select a drive. Use the information from Table 16-2 to set the appropriate motor on bit.

Table 16-3. Drive Select

BIT 0	BIT 1	SELECTED DRIVE
0	0	$\overline{\text{DRVSEL0}}$ — Selects disk drive A, the first floppy disk drive mounted in the computer.
1	0	$\overline{\text{DRVSEL1}}$ — Selects disk drive B, the second floppy disk drive mounted in the computer.
0	1	$\overline{\text{DRVSEL2}}$ — Selects a third floppy disk drive, if installed.
1	1	$\overline{\text{DRVSEL3}}$ — Selects a fourth floppy disk drive, if installed.

All eight bits of the digital output register are cleared (set to 0) when the computer is first turned on and each time an I/O reset occurs. Each bit is set to the corresponding input when the clock signal makes a transition from low to high. The clock signal for this register is developed by the chip select signal, address line 2, and the IOW signal.

Main Status Register (Port 3F4H)

The main status register is an 8-bit read-only register located at port address 3F4H. This register contains the status information for the floppy disk controller and may be read at any time. However, it is recommended that the CPU wait 12 μ s each time it reads the main status register. The wait allows the data direction and request for master bits to change and settle. Each bit in the main status register is described in Table 16-4.

Mass Storage

Table 16-4. Main Status Register

BIT	FUNCTION	DESCRIPTION
0	Floppy drive 0 busy	This bit is set (1) when disk drive A is busy (in the seek mode). The controller will not accept read or write commands for any disk drive.
1	Floppy drive 1 busy	This bit is set (1) when disk drive B is busy (in the seek mode). The controller will not accept read or write commands for any disk drive.
2	Floppy drive 2 busy	This bit is set (1) when disk drive C is busy (in the seek mode). The controller will not accept read or write commands for any disk drive.
3	Floppy drive 3 busy	This bit is set (1) when disk drive D is busy (in the seek mode). The controller will not accept read or write commands for any disk drive.
4	Controller busy	This bit is set (1) when a read or write command is in process. The controller will not accept any other command.
5	Execution mode	This bit is set (1) during the execution phase of a non-DMA operation. The bit operates only in non-DMA mode.
6	Data direction	This bit is set (1) when data is transferred from the data register stack to the system data bus. It is clear (0) when data is transferred from the system data bus to the data register stack.
7	Request for master	This bit is set (1) when the data register is ready for a transfer operation. It is cleared (0) when the data register is not ready.

Floppy Control Register (Port 3F7H)

Bits 0 and 1 of the floppy control register set the data transfer rate in kilobits per second. Bits 2 through 7 are reserved. Table 16-5 defines the encoding for the data transfer rates.

Table 16-5. Floppy Control Register

BIT 0	BIT 1	DATA TRANSFER RATE
0	0	500 kbps
1	0	300 kbps ¹
0	1	250 kbps
1	1	125 kbps ²

NOTES

1. The controller must be programmed for MFM encoding.
2. The controller must be programmed for FM encoding.

Data Register (Port 3F5H)

The data register port provides sequential access to a stack of 8-bit registers used to program the functions of the controller, temporarily hold the data being read from or written to the disk, and provide additional status reports. The sequence and data used with the data register stack depends on the programmed instruction. Only one register is presented to the data bus at a time. Four of these registers are status registers (ST0-ST3). These status registers are only available during the result phase of a command and may only be read after a command has been executed. All of the command bytes (usually 9) must be written to the data register port before the execution phase can begin. All of the result bytes (usually 7) must be read from this register before another command can be executed. Table 16-6 provides bit descriptions for each of the status registers.

Mass Storage

Table 16-6. Floppy Data Registers ST0-ST3

REGISTER	BIT	FUNCTION	DESCRIPTION
ST0	0-1	Drive select	These bits indicate which drive interrupted the CPU. See Table 16-3.
	2	Head address	This bit indicates the state of the read/write head at an interrupt.
	3	Not ready	This bit is set when the drive is not ready and a read or write command has been issued. It is also set if a read or write command has been issued for side 1 of a single-sided disk.
	4	Equipment check	This bit is set when a fault signal has been received from the controller or if the track 0 signal is not received after 77 step pulses during the execution of the recalibrate command.
	5	Seek end	This bit is set upon completion of a seek command.
	6-7	Interrupt code	These bits indicate whether a command was completed and executed properly, started but not executed properly, or not started.
ST1	0	Missing address	This bit is set if the controller cannot find an address mark or a deleted address mark. The missing address mark bit in register ST2 is also set when this occurs.
	1	Write protect	This bit is set when a write protect signal is detected during a write data, write deleted data, or format a cylinder command.

Table 16-6 (continued). Floppy Data Registers ST0-ST3

REGISTER	BIT	FUNCTION	DESCRIPTION
	2	No data	This bit is set if the controller cannot find the sector specified by the internal data register during a read, write, or scan command. It is also set if the ID field cannot be read without an error during a read ID command. It can also be set during a read cylinder command if the starting sector cannot be found.
	3	Not used	This bit is always cleared.
	4	Overrun	This bit is set if the controller is not serviced by the processor or DMA within a specified length of time.
	5	Data error	This bit is set when a CRC error is detected in either the ID or data field.
	6	Not used	This bit is always cleared.
	7	End of cylinder	This bit is set when the controller attempts to access a sector beyond the last sector in a cylinder.
ST2	0	Missing address	This bit is set in conjunction with the missing address bit of ST0 when the controller cannot find a data address mark or a deleted data address mark during a read command.
	1	Bad cylinder	When the contents of a cylinder is FFH and does not match the data stored in the internal data register, this bit is set.

Mass Storage

Table 16-6 (continued). Floppy Data Registers ST0-ST3

REGISTER	BIT	FUNCTION	DESCRIPTION
	2	Scan not satisfied	This bit is set when the controller cannot find a sector that meets the condition specified in a scan command.
	3	Scan equal hit	This bit is set during the execution of a scan command if the equal condition is met.
	4	Wrong cylinder	This bit is set when the contents of a cylinder do not match the contents of the internal data register.
	5	Data error	When the controller detects a CRC error in the data field, this bit is set.
	6	Control mark	This bit is set when the controller detects a deleted address mark during the execution of a read or scan command.
	7	Not used	This bit is always cleared.
	0-1	Drive select	These bits reflect the drive select signals sent to the drive. See Table 16-3.
ST3	2	Head address	This bit reflects the status of the side select signal sent to the drive.
	3	Two-side	This bit reflects the status of the two-sided signal from the drive.
	4	Track 0	This bit reflects the status of the track 0 signal from the drive.
	5	Ready	This bit reflects the status of the ready signal from the drive.

Table 16-6 (continued). Floppy Data Registers ST0-ST3

REGISTER	BIT	FUNCTION	DESCRIPTION
	6	Write protect	This bit reflects the status of the write protect signal from the drive.
	7	Fault	This bit reflects the status of the fault signal from the drive.

FDC Pinout

The 765 FDC requires a +5 VDC voltage for normal operation. The clock signal used for this device is a single phase 8 MHz square wave. The pinout for this device is illustrated in Figure 16-2. Table 16-7 describes the signals used by the 765 FDC.

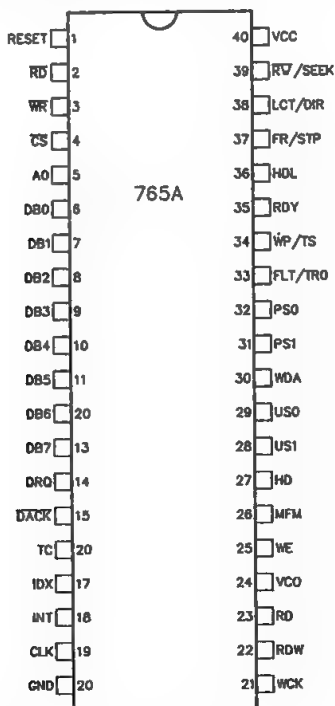


Figure 16-2. 765 FDC Pinout

Mass Storage

Table 16-7. 765 FDC Signal Descriptions

PIN NUMBER	SIGNAL NAME	DESCRIPTION
1	RESET	Reset. This input resets the output lines and places the FDC in an idle state. If the RDY input is held high during RESET, an interrupt will be generated within 1.024 ms. To clear the interrupt, use the sense interrupt status command.
2	\overline{RD}	Read. A low on this input allows data transfers from FDC to bus. A high (1) on the \overline{CS} line disables this input.
3	\overline{WR}	Write. A low on this input allows data transfers from the bus to the FDC. A high (1) on the \overline{CS} line disables this input.
4	\overline{CS}	Chip select. This active-low input selects the FDC and enables \overline{RD} , \overline{WR} , and A0.
5	A0	Data/status register selection. A high on this input (A0 = 1) selects the data register contents and a low (A0 = 0) selects the status register contents for transfer to the data bus.
6–13	D0–D7	8-bit bidirectional data bus. These lines are disabled when \overline{CS} is held high.
14	DRQ	Data DMA request. The FDC requests a DMA transfer by asserting this output (DRQ = 1).
15	\overline{DACK}	DMA acknowledge. This input line is held low during an active DMA cycle, while the FDC is performing a DMA transfer.
16	TC	Terminal count. This output line indicates DMA cycle completion when high. It also terminates data transfers during read, write, and scan commands in either DMA or non-DMA mode.
17	IDX	Index. A high on this input indicates the beginning of a disk track.

Table 16-7 (continued). 765 FDC Signal Descriptions

PIN NUMBER	SIGNAL NAME	DESCRIPTION
18	INT	Interrupt. This output line is the FDC interrupt request line.
19	CLK	Clock. Single-phase 8 MHz square wave clock.
20	GND	Ground. DC power return.
21	WCK	Write clock. This input line sets the datawrite rate for the disk drive. MFM mode uses a 1MHz, 250 ns pulse; FM mode uses a 500kHz, 250ns pulse.
22	RDW	Read data window. This input line is connected to the phase-locked loop (PLL) and is used to sample data from disk.
23	RDD	Read data. This is the serial data and clock input line used for reading data from the drive.
24	VCO/Sync	VCO/sync. This output disables the VCO signal from the PLL when low and enables it when high.
25	WE	Write enable. This output line enables writing data to disk.
26	MFM	MFM mode. This output line reflects the disk encoding format: 1 = MFM, 0 = FM.
27	HD	Head select. This output line reflects the head selected: 1 = head 1, 0 = head 0.
28,29	US1, US0	Unit select. This output line reflects the disk drive unit selected.
30	WDA	Write data. This serial clock and data output line transfers data to disk.
31,32	PS1, PS0	Precompensation (preshift). These outputs reflect the write precompensation status for MFM mode. They indicate early, late, or normal timing.

Mass Storage

Table 16-7 (continued). 765 FDC Signal Descriptions

PIN NUMBER	SIGNAL NAME	DESCRIPTION
33	FLT/TR0	Fault/track 0. This input line is used for fault sensing in read/write mode and track 0 indication in seek mode.
34	WP/TS	Write protect/two side. This input line senses write protection in read/write mode and two-sided media in seek mode.
35	RDY	Ready. This input line receives the disk drive ready signal.
36	HDL	Head load. This output line causes the read/write heads to contact the disk.
37	FR/STP	Fault reset/step. This output line resets the fault flip-flop in the drive when in read/write mode and provides step pulses in seek mode. A fault reset pulse must be issued prior to asserting HDL for read/write operations.
38	LCT/DIR	Low current/direction. This output line lowers write current on inner tracks in read/write mode and determines seek direction in seek mode.
39	$\overline{\text{RW/SEEK}}$	Read write/seek. This output line selects either read/write or seek mode: seek = 1, read/write mode = 0.
40	Vcc	+5V DC power.

Programming Floppy Disk Operations

Before the FDC can carry out a disk operation, the digital output register and the floppy control register must be initialized. For example, a data byte consisting of the bit pattern 00000000 (00H), which selects a data transfer rate of 500 kbps, must be output to port 3F7H. Then, a data byte consisting of the bit pattern 00011100 (1CH) which, selects drive A, enables the DMA and interrupt circuits, and turns on disk drive A's motor, must be output to port 3F2H.

NOTE: Bit 2 of the digital output register should only be cleared if a serious error occurs. When the controller is reset, the disk drives must be recalibrated, moving each drive's head to track 0.

The following sequence represents a typical series of steps that could be used in programming a disk operation:

1. Specify head settling time and amount of data to be transferred.
2. Specify the data transfer rate and the recording mode (FM or MFM).
3. Select a drive and turn on the motor.
4. Wait for the motor to come to speed.
5. Send the command instruction set to the controller.
6. Wait for the interrupt that indicates the completion of the operation.
7. Initialize the DMA to move data to or from memory.
8. Send the proper instruction to the controller to read or write data.
9. Wait for the interrupt that indicates the completion of the operation.

Mass Storage

10. Read the status report from the controller and analyze it to make sure the operation was performed correctly.
11. Turn off the motor.

These steps represent a simplified version of a basic disk access operation, such as reading or writing a sector. Due to the structure of the disk, reading or writing a sector is not quite this simple. In order to read or write a specific sector, first read the disk directory to locate the file. Next, use the file allocation tables to locate the desired sector on the disk. Third, move the heads to the track and locate the sector, and then perform the read or write operation. The software interrupts contain all the necessary routines to perform these complex operations. Refer to Chapter 10 for complete programming information on the disk drive interrupts. In addition, refer to the Programmer's Utility Package for a description of the organization of the MS-DOS disk.

Floppy Disk Controller Commands

The FDC can execute 15 different commands. Each command is initiated by outputting a multibyte instruction to the floppy data register at port 3F5H. The controller then executes the commands and returns a multibyte result report to the floppy data register. These result bytes must be read by the microprocessor before another command can be initiated.

For more information on programming this controller, refer to the NEC's μ PD765A/7265 data sheet. The available controller commands include the following:

- Reading and writing one or more sectors
- Reading an entire track
- Locating the position of the head on a track
- Formatting a track
- Scanning and comparing written data against memory
- Seeking to a track
- Calibrating (synchronizing) head-to-track information
- Checking interrupt and drive status
- Seeking to a specified track.

Table 16-8 describes the 15 commands available in the FDC.

Table 16-8. Floppy Disk Controller Commands

COMMAND	DESCRIPTION
Read data	Reads data from one or more sectors of a two-track cylinder.
Read deleted data	Allows a sector marked with a deleted data mark to be read.
Write data	Writes data to one or more sectors of a one- or two-track cylinder.
Write deleted data	Same as a write data command except that deleted data marks are written instead of normal data marks.
Read a track	Reads the contents of an entire track.
Read sector ID	Reads the values of the first ID field it is able to read.
Format a track	Allows an entire track to be formatted.
Scan equal	Reads the data on the disk and compares it to the data supplied by the processor. If the two sets of data are equal, the operation terminates.
Scan low or equal	Reads the data on the disk and compares it to the data supplied by the processor. If the disk data is of equal or lesser value, the operation terminates.

Mass Storage

Table 16-8 (continued). Floppy Disk Controller Commands

COMMAND	DESCRIPTION
Scan high or equal	Reads the data on the disk and compares it to the data supplied by the processor. If the disk data is of equal or greater value, the operation terminates.
Recalibrate	Causes the head of the selected drive to be moved to track 0. The internal track counter is cleared for this drive.
Sense interrupt status	Allows the user to sense interrupts that result during seek and recalibrate commands, which have no result phase.
Specify	Sets the initial values of each of three internal timers. These are the head unload time, the step rate time, and the head load time.
Drive status	Reports the status of the disk drives.
Seek	Causes the disk drive to seek to the specified cylinder. The specified drive's track counter is incremented or decremented according to the number of step pulses sent to the drive.

Most of the commands begin with a byte that initializes the controller for the operation that follows. This byte allows deleted address marks to be skipped, selects FM or MFM encoding, and allows use of multi-track mode. This byte is usually followed by a head and drive select byte. Table 16-9 lists each of the bits used for these command bytes and describes their function. The bit position for the command bytes is described with each instruction.

Table 16-9. Controller Command Bits

BIT	DESCRIPTION
Skip bit	If set (1), this bit causes the operation to skip the deleted data address mark.
FM/MFM bit	If clear (0), the controller is placed in the FM read/write mode. If set (1), the controller is placed in the MFM read/write mode.
Multitrack bit	If set (1), the multitrack mode is in effect. At the end of side 0, side 1 will be used automatically.
Head number bit	This is the same as the head address and will be set (1) for side 1 or clear (0) for side 0.
Drive unit number bits	These two bits specify the drive unit number: unit 0 = drive A, unit 1 = drive B, unit 2 = drive C, and unit 3 = drive D.

Read data — This command uses nine command bytes and returns seven result bytes. It reads data from one or more sectors. Each of the command and status bytes are described in Tables 16-9 and 16-10.

Table 16-10. Read Data Command

PHASE	BYTE	DESCRIPTION
Command	1	Write the first set of command codes. Bits 0 – 4 = 06H; bit 5 = skip bit; bit 6 = FM/MFM bit; bit 7 = multitrack bit. Refer to Table 16-9.
	2	Write the second set of command codes. Bits 0 – 1 = drive unit number (0 – 3); bit 2 = head number; bits 3 – 7 = not used. Refer to Table 16-9.
	3	Write the cylinder (track) number.

Mass Storage

Table 16-10 (continued). Read Data Command

	4	Write the head address. This value is equal to the value of the head number (0 or 1), as defined in Table 16-6.
	5	Write the sector number to be read.
	6	Write the number of bytes per sector.
	7	Write the end of track, which is the final sector number to be read on a cylinder.
	8	Write the gap length, which is used for synchronization in read operations.
	9	Write the data length. If the bytes per sector value (byte 6) is 0, this value is the data length to be read from or written to each sector.
Result	1	Read ST0. Refer to Table 16-6.
	2	Read ST1. Refer to Table 16-6.
	3	Read ST2. Refer to Table 16-6.
	4	Read cylinder (track) number.
	5	Read head number.
	6	Read sector number.
	7	Read bytes per sector.

Read deleted data — This command uses nine command bytes and returns seven result bytes. It reads data from a sector that has been marked with a deleted data mark, as described in Table 16-11. With the exception of the first set of command codes, this command is identical to the read data command.

Table 16-11. Read Deleted Data Command

PHASE	BYTE	DESCRIPTION
Command	1	Write the first set of command codes. Bits 0 – 4 = 0CH; bit 5 = skip bit; bit 6 = FM/MFM bit; bit 7 = multitrack bit. Refer to Table 16-9.
	2	Write the second set of command codes. Bits 0 – 1 = drive unit number (0 – 3); bit 2 = head number; bits 3 – 7 = not used. Refer to Table 16-9.
	3	Write the cylinder (track) number.
	4	Write the head address. This value is equal to the value of the head number (0 or 1), as defined in Table 16-9.
	5	Write the sector number to be read.
	6	Write the number of bytes per sector.
	7	Write the end of track, the final sector number to be read on a cylinder.
	8	Write the gap length, which is used for synchronization in read operations.
	9	Write the data length. If the bytes per sector value (byte 6) is 0, this value is the data length to be read from or written to each sector.

Mass Storage

Table 16-11 (continued). Read Deleted Data Command

PHASE	BYTE	DESCRIPTION
Result	1	Read ST0. Refer to Table 16-6.
	2	Read ST1. Refer to Table 16-6.
	3	Read ST2. Refer to Table 16-6.
	4	Read cylinder (track) number.
	5	Read head number.
	6	Read sector number.
	7	Read bytes per sector.

Write data — This command uses nine command bytes and returns seven result bytes. It writes data to one or more sectors of a two-track cylinder, as described in Table 16-12.

Table 16-12. Write Data Command

PHASE	BYTE	DESCRIPTION
Command	1	Write the first set of command codes. Bits 0 – 4 = 05H; bit 5 = 0; bit 6 = FM/MFM bit; bit 7 = multitrack bit. Refer to Table 16-9.
	2	Write the second set of command codes. Bits 0 – 1 = drive unit number (0 – 3); bit 2 = head number; bits 3 – 7 = not used. Refer to Table 16-9.
	3	Write the cylinder (track) number.
	4	Write the head address. This value is equal to the value of the head number (0 or 1), as defined in Table 16-9.
	5	Write the sector number to be read.

Table 16-12 (continued). Write Data Command

PHASE	BYTE	DESCRIPTION
	6	Write the number of bytes per sector.
	7	Write the end of track, the final sector number to be read on a cylinder.
	8	Write the gap length, which is used for synchronization in read operations.
	9	Write the data length. If the bytes per sector value (byte 6) is 0, this value is the data length to be read from or written to each sector.
Result	1	Read ST0. Refer to Table 16-6.
	2	Read ST1. Refer to Table 16-6.
	3	Read ST2. Refer to Table 16-6.
	4	Read cylinder (track) number.
	5	Read head number.
	6	Read sector number.
	7	Read bytes per sector.

Write deleted data — This command uses nine command bytes and returns seven result bytes. It writes data with a deleted data mark as described in Table 16-13. With the exception of the first set of command codes, this command is identical to the write data command.

Mass Storage

Table 16-13. Write Deleted Data Command

PHASE	BYTE	DESCRIPTION
Command	1	Write the first set of command codes. Bits 0 – 4 = 09H; bit 5 = 0; bit 6 = FM/MFM bit; bit 7 = multitrack bit. Refer to Table 16-9.
	2	Write the second set of command codes. Bits 0 – 1 = drive unit number (0 – 3); bit 2 = head number; bits 3 – 7 = not used. Refer to Table 16-9.
	3	Write the cylinder (track) number.
	4	Write the head address. This value is equal to the value of the head number (0 or 1), as defined in Table 16-9.
	5	Write the sector number to be read.
	6	Write the number of bytes per sector.
	7	Write the end of track, the final sector number to be read on a cylinder.
	8	Write the gap length, which is used for synchronization in read operations.
	9	Write the data length. If the bytes per sector value (byte 6) is 0, this value is the data length to be read from or written to each sector.
Result	1	Read ST0. Refer to Table 16-6.
	2	Read ST1. Refer to Table 16-6.
	3	Read ST2. Refer to Table 16-6.
	4	Read cylinder (track) number.
	5	Read head number.
	6	Read sector number.
	7	Read bytes per sector.

Mass Storage

Read a track — This command uses nine command bytes and returns seven result bytes. It reads the contents of an entire track from the index (timing) hole to the end of the track, as described in Table 16-14.

Table 16-14 (continued). Read a Track Command

PHASE	BYTE	DESCRIPTION
Command	1	Write the first set of command codes. Bits 0 – 4 = 02H; bit 5 = skip bit; bit 6 = FM/MFM bit; bit 7 = 0. Refer to Table 16-9.
	2	Write the second set of command codes. Bits 0 – 1 = drive unit number (0 – 3); bit 2 = head number; bits 3 – 7 = not used. Refer to Table 16-9.
	3	Write the cylinder (track) number.
	4	Write the head address. This value is equal to the value of the head number (0 or 1), as defined in Table 16-9.
	5	Write the sector number to be read.
	6	Write the number of bytes per sector.
	7	Write the end of track, the final sector number to be read on a cylinder.
	8	Write the gap length, which is used for synchronization in read operations.
	9	Write the data length. If the bytes per sector value (byte 6) is 0, this value is the data length to be read from or written to each sector.
Result	1	Read ST0. Refer to Table 16-6.
	2	Read ST1. Refer to Table 16-6.
	3	Read ST2. Refer to Table 16-6.

Mass Storage

Table 16-14 (continued). Read a Track Command

PHASE	BYTE	DESCRIPTION
	4	Read cylinder (track) number.
	5	Read head number.
	6	Read sector number.
	7	Read bytes per sector.

Read sector ID — This command uses two command bytes and returns seven result bytes. It reads the values of the first ID field it is able to read, as described in Table 16-15.

Table 16-15. Read Sector ID Command

PHASE	BYTE	DESCRIPTION
Command	1	Write the first set of command codes. Bits 0 – 4 = 0AH; bit 5 = 0; bit 6 = FM/MFM bit; bit 7 = 0. Refer to Table 16-9.
	2	Write the second set of command codes. Bits 0 – 1 = drive unit number (0 – 3); bit 2 = head number; bits 3 – 7 = not used. Refer to Table 16-9.
Result	1	Read ST0. Refer to Table 16-6.
	2	Read ST1. Refer to Table 16-6.
	3	Read ST2. Refer to Table 16-6.
	4	Read cylinder (track) number.
	5	Read head number.
	6	Read sector number.
	7	Read bytes per sector.

Mass Storage

Format a track — This command uses six command bytes and returns seven result bytes. It formats an entire track, as described in Table 16-16.

Table 16-16. Format a Track Command

PHASE	BYTE	DESCRIPTION
Command	1	Write the first set of command codes. Bits 0 – 4 = 0DH; bit 5 = 0; bit 6 = FM/MFM bit; bit 7 = 0. Refer to Table 16-9.
	2	Write the second set of command codes. Bits 0 – 1 = drive unit number (0 – 3); bit 2 = head number; bits 3 – 7 = not used. Refer to Table 16-9.
	3	Write the number of bytes per sector.
	4	Write the number of sectors per track.
	5	Write the number of bytes in gap 3. During read/write commands, this value determines the time that the sync signal will be active.
	6	Write the pattern to be used in the data area of each sector.
Result	1	Read ST0. Refer to Table 16-6.
	2	Read ST1. Refer to Table 16-6.
	3	Read ST2. Refer to Table 16-6.
	4	Read cylinder (track) number.
	5	Read head number.
	6	Read sector number.
	7	Read bytes per sector.

Mass Storage

Scan equal — This command uses nine command bytes and returns seven result bytes. It reads the data on the disk and compares it with the data supplied by the processor as described in Table 16-17. If the two sets of data are equal, the operation terminates.

Table 16-17. Scan Equal Command

PHASE	BYTE	DESCRIPTION
Command	1	Write the first set of command codes. Bits 0 – 4 = 11H; bit 5 = skip bit; bit 6 = FM/MFM bit; bit 7 = multitrack bit. Refer to Table 16-9.
	2	Write the second set of command codes. Bits 0 – 1 = drive unit number (0 – 3); bit 2 = head number; bits 3 – 7 = not used. Refer to Table 16-9.
	3	Write the cylinder (track) number.
	4	Write the head address. This value is equal to the value of the head number (0 or 1), as defined in Table 16-9.
	5	Write the sector number to be read.
	6	Write the number of bytes per sector.
	7	Write the end of track, the final sector number to be read on a cylinder.
	8	Write the gap length, which is used for synchronization in read operations.
	9	Write the sector timing pattern. If this byte = 1, the sectors are contiguous. If this byte = 2, the sectors alternate.
Result	1	Read ST0. Refer to Table 16-6.
	2	Read ST1. Refer to Table 16-6.
	3	Read ST2. Refer to Table 16-6.

Table 16-17 (continued). Scan Equal Command

PHASE	BYTE	DESCRIPTION
	4	Read cylinder (track) number.
	5	Read head number.
	6	Read sector number.
	7	Read bytes per sector.

Scan low or equal — This command uses nine command bytes and returns seven result bytes. It reads the data on the disk and compares it with the data supplied by the processor, as described in Table 16-18. If the disk data is of equal or of lesser value, the operation terminates.

Table 16-18. Scan Low or Equal Command

PHASE	BYTE	DESCRIPTION
Command	1	Write the first set of command codes. Bits 0 – 4 = 19H; bit 5 = skip bit; bit 6 = FM/MFM bit; bit 7 = multitrack bit. Refer to Table 16-9.
	2	Write the second set of command codes. Bits 0 – 1 = drive unit number (0 – 3); bit 2 = head number; bits 3 – 7 = not used. Refer to Table 16-9.
	3	Write the cylinder (track) number.
	4	Write the head address. This value is equal to the value of the head number (0 or 1), as defined in Table 16-9.
	5	Write the sector number to be read.
	6	Write the number of bytes per sector.
	7	Write the end of track, the final sector number to be read on a cylinder.

Mass Storage

Table 16-18 (continued). Scan Low or Equal Command

PHASE	BYTE	DESCRIPTION
	8	Write the gap length, which is used for synchronization in read operations.
	9	Write the sector timing pattern. If this byte = 1, the sectors are contiguous. If this byte = 2, the sectors alternate.
Result	1	Read ST0. Refer to Table 16-6.
	2	Read ST1. Refer to Table 16-6.
	3	Read ST2. Refer to Table 16-6.
	4	Read cylinder (track) number.
	5	Read head number.
	6	Read sector number.
	7	Read bytes per sector.

Scan high or equal — This command uses nine command bytes and returns seven result bytes. It reads the data on the disk and compares it with the data supplied by the processor, as described in Table 16-19. If the disk data is of equal or greater value, the operation terminates.

Table 16-19 Scan High or Equal Command

PHASE	BYTE	DESCRIPTION
Command	1	Write the first set of command codes. Bits 0 – 4 = 1DH; bit 5 = skip bit; bit 6 = FM/MFM bit; bit 7 = multitrack bit. Refer to Table 16-9.
	2	Write the second set of command codes. Bits 0 – 1 = drive unit number (0 – 3); bit 2 = head number; bits 3 – 7 = not used. Refer to Table 16-9.
	3	Write the cylinder (track) number.
	4	Write the head address. This value is equal to the value of the head number (0 or 1), as defined in Table 16-9.
	5	Write the sector number to be read.
	6	Write the number of bytes per sector.
	7	Write the end of track, the final sector number to be read on a cylinder.
	8	Write the gap length, which is used for synchronization in read operations.
	9	Write the sector timing pattern. If this byte = 1, the sectors are contiguous. If this byte = 2, the sectors alternate.
Result	1	Read ST0. Refer to Table 16-6.
	2	Read ST1. Refer to Table 16-6.
	3	Read ST2. Refer to Table 16-6.
	4	Read cylinder (track) number.
	5	Read head number.
	6	Read sector number.
	7	Read bytes per sector.

Mass Storage

Recalibrate — This command uses two command bytes and does not return any result bytes. It causes the head of the selected drive to be moved to track 0. The internal track counter for the drive is cleared. Refer to Table 16-20 for the description of the two command bytes. A sense interrupt command must be sent following this command or the FDC will consider the next command to be invalid.

Table 16-20. Recalibrate Command

PHASE	BYTE	DESCRIPTION
Command	1	Write the first command code. Bits 0 – 7 = 07H.
	2	Write the second set of command codes. Bits 0 – 1 = drive unit number (0 – 3); bits 2 – 7 = not used. Refer to Table 16-9.

Sense interrupt status — This command uses one command byte and returns two result bytes. It allows the user to sense interrupts that result during seek and recalibrate commands, as described in Table 16-21. It is used in conjunction with seek and recalibrate commands, since they have no result phase.

Table 16-21. Sense Interrupt Status Command

PHASE	BYTE	DESCRIPTION
Command	1	Write the command code. Bits 0 – 7 = 08H.
Result	1	Read ST0. Refer to Table 16-6.
	2	Read cylinder (track) number.

Mass Storage

Specify — This command uses three command bytes. It sets the initial values of each of three internal timers, as described in Table 16-22. These are the head unload time, the step rate time, and the head load time.

Table 16-22. Specify Command

PHASE	BYTE	DESCRIPTION
Command	1	Write the command code. Bits 0–7 = 03H.
	2	Write the head unload time and step rate time. Bits 0–3 = head unload time in 16 ms increments. Bits 4–7 = step rate time in 1 ms increment.
Result	1	Read ST3. Refer to Table 16-6.

Drive Status — This command uses two command bytes and returns one result byte. It reports the status of the specified disk drive, as described in Table 16-23.

Table 16-23. Drive Status Command

PHASE	BYTE	DESCRIPTION
Command	1	Write the first command code. Bits 0–7 = 04H.
	2	Write the second set of command codes. Bits 0–1 = drive unit number (0–3); bit 2 = head number; bits 3–7 = not used. Refer to Table 16-9.
Result	1	Read ST3. Refer to Table 16-6.

Mass Storage

Seek — This command uses three command bytes. It causes the specified disk drive to seek to the specified cylinder, as described in Table 16-24. The specified drive's track counter is incremented or decremented according to the number of step pulses sent to the drive. A sense interrupt status command must be sent after this command or the FDC will consider the next command invalid.

Table 16-24. Seek Command

PHASE	BYTE	DESCRIPTION
Command	1	Write the first command code. Bits 0 – 7 = 0FH.
	2	Write the second set of command codes. Bits 0 – 1 = drive unit number (0 – 3); bit 2 = head number; bits 3 – 7 = not used. Refer to Table 16-9.
	3	Write the new cylinder number to which the heads are to be stepped.

NOTE: — Any other command codes that are sent to the data register will result in a NOP operation and one status byte with a value of 80H will be returned.

Hard Disk Control

This computer contains all circuitry necessary to support the internal hard disk drive. The hard disk controller contains firmware that responds to control signals provided by the BIOS firmware. Chapter 10 discusses using the BIOS interrupts. Custom circuits provide PC-compatible hard disk control, including timing, automatic head and cylinder switching, automatic seek and verify of each data transfer, data error sensing and correction, sector interleave, enhanced step rates, and removable cartridge drive support.

Data is transferred to the hard disk at a rate of 5 Mbps, in 16-bit words. The internal controller can also select an 8-bit mode, allowing four additional bytes to be transferred for long-word read and write commands. A 512-byte sector buffer is provided to prevent data overruns during transfer operations.

Hard Disk Drive Control Registers

Eight task file registers and two auxiliary registers control and provide status reports for the hard disk drive controller. Table 16-25 lists the primary and secondary port addresses for these registers. The table also illustrates the read and write operation for each register.

Table 16-25. Hard Disk Drive Control Register Port Addresses

PRIMARY PORT ADDRESS	SECONDARY PORT ADDRESS	READ OPERATION REGISTER	WRITE OPERATION REGISTER
1F0H	170H	Data	Data
1F1H	171H	Error	Write precomp
1F2H	172H	Sector count	Sector count
1F3H	173H	Sector number	Sector number
1F4H	174H	Cylinder low	Cylinder low
1F5H	175H	Cylinder high	Cylinder high
1F6H	176H	Drive & head	Drive & head
1F7H	177H	Status	Command
3F6H	376H	Not used	Fixed disk
3F7H	377H	Digital input	Digital output

NOTE: The digital output register and bit 7 of the digital input register are used in floppy disk drive operations.

Data Register (Port 1F0H)

The data register is used in parallel read and write operations. It provides a 16-bit path to the sector buffer. Data bytes pass through this buffer to prevent overruns during read and write operations. An internal controller allows the register to select 8-bit mode, allowing the additional 4 bytes of ECC code to be transferred in a read or write long command.

Mass Storage

Error Register (Port 1F1H)

The error register is a read-only register which returns a 1-byte code giving error information about the last command. Two sets of codes are used. The first set, listed in Table 16-26, appears following the execution of a diagnostic command. The second set, listed in Table 16-27, appears following the execution of any other command.

Following the execution of a diagnostic command, the controller is in diagnostic mode. In this mode the hex value of the register contents determines the error condition. In normal operation mode, each bit of the register indicates a particular error condition. The contents of this register are valid only when the error bit of the status register is set.

Table 16-26. Diagnostic Mode Error Register Codes

CODE	ERROR
01H	No error
02H	Controller error
03H	Sector buffer error
04H	ECC device error
05H	Control processor error

Table 16-27. Operation Mode Error Register Codes

BIT	FUNCTION	DESCRIPTION
0	Data address mark not found	This bit is set if the data address mark is not found in the 16 bytes following the ID field. If the controller is programmed for retry, 16 passes are made before the bit is set. If the controller is not programmed for retry, only one pass is made.
1	Track 0	This bit is set during a restore command if the track 0 signal is not set within 2047 step pulses.

Table 16-27 (continued). Operation Mode Error Register Codes

BIT	FUNCTION	DESCRIPTION
2	Aborted command	This bit is set if an invalid command is received by the controller. It is also set if a drive not ready, no seek complete, or write fault signal are received from the drive.
3	Cartridge changed	This bit reflects the status of the cartridge changed line of the drive. It must be cleared using the acknowledge cartridge changed command before any other command can be executed.
4	ID not found	This bit is set if the desired ID field is not found within 2 revolutions of search. When the controller has been programmed for retry, 16 search attempts are made. Also, an automatic seek to the correct cylinder is issued if the heads are located on the wrong cylinder. If the controller is not programmed for retry, no automatic seek is issued if the heads are located on the wrong cylinder. The error bit is set following 2 revolutions of search.
5	Write protected	This bit is only used with removable cartridge drives. If the cartridge has been write protected, the bit is set.
6	Data ECC	This bit is set if a noncorrectable data ECC error is detected.
7	Bad block detect	This bit is set when a bad block mark is read from the ID field. When this bit is set, any read or write operation is aborted.

Write-Precompensation Register (Port 1F1H)

The write-precompensation register is an 8-bit write only register. The hex value written to this register is multiplied by 4 by the controller and the resultant value is used as the first precompensated cylinder number.

Sector Count Register (Port 1F2H)

The sector count register is an 8-bit register used in conjunction with the sector number register. It specifies the number of sectors to be transferred during a read or write operation. Multiple sector operations are supported and may cross tracks and cylinders. The appropriate drive characteristics must be loaded using the set drive parameter command before a multiple sector transfer can be executed. During multiple sector operations, this register is decremented and the sector number register is incremented after each successful sector transfer. When a value of zero is placed in the sector count register, a transfer of 256 sectors is specified.

Sector Number Register (Port 1F3H)

This 8-bit register is used during read, write, and verify operations. It must be loaded with the logical starting sector number prior to execution of the command. Upon completion of the command, the register is updated with the last sector read or written. If an error is detected, upon termination of the command, the register is loaded with the sector in error.

Cylinder Low and Cylinder High Registers (Ports 1F4H and 1F5H)

This pair of 8-bit registers are used during read, write, seek, and format operations. The cylinder low register (port 1F4H) is loaded with the low-byte hex value of the starting cylinder for the operation. The cylinder high register (port 1F5H) contains the starting cylinder high-byte hex information. Upon completion of the operation, the registers are updated and contain current cylinder information.

Drive and Head Register (Port 1F6H)

This 8-bit write-only register is used to program drive and head select for most operations. It is also used with the set parameters command to indicate the maximum number of heads for each drive. Table 16-28 provides coding information for this register. The controller supports a maximum of two hard disk drives with a maximum of 16 heads each.

Table 16-28. Drive and Head Register Codes

BIT	FUNCTION	DEFINITION
0-3	Head select	Enter a hex value ranging from 0-F. The value indicates the head selected for most operations. The maximum number of heads for the drive indicated by bits 4-7 for the set parameters command.
4-7	Drive select	Enter the bit pattern 1010 (AH) to select or set the parameters for drive 0. Enter the bit pattern 1011 (BH) to select or set parameters for drive 1.

Status Register (Port 1F7H)

This 8-bit read-only register contains a 1-byte code that indicates the status of the drive and the controller. Reading this register clears the hard disk interrupt. Table 16-29 provides the decoding information for the register.

Table 16-29. Status Register Codes

BIT	FUNCTION	DEFINITION
0	Error	This bit is set when an error is encountered while executing a command. It indicates that the command was terminated. When this bit is set, the microprocessor should read the error register to determine the nature of the error.

Mass Storage

Table 16-29 (continued). Status Register Codes

BIT	FUNCTION	DEFINITION
1	Index	This bit reflects the state of the index signal from the drive.
2	Corrected data	This bit is set during a read command when the data in the buffer awaiting transfer has been corrected using ECC. This bit does not cause a multiple sector read command to abort if a correctable data error is encountered, but does remain set until a new command is issued.
3	Data request	This bit is a signal to the microprocessor that the controller is ready for data to be read from or written to the data register. When the appropriate amount of data has been transferred, the bit is cleared.
4	Seek complete	This bit reflects the state of the seek complete signal from the drive.
5	Write fault	This bit reflects the state of the write fault signal from the drive.
6	Drive ready	This bit reflects the state of the drive ready signal from the drive.
7	Busy	This bit is set while the controller is executing a command. While this bit is set, the contents of bits 1, 4, 5, and 6 are valid. However, all other bits, and all other registers, are considered invalid. Check the status of this bit prior to reading any other registers.

Command Register (Port 1F7H)

The 8-bit code written to this register provides the controller with the commands to be executed. The controller is capable of executing thirteen commands. Prior to writing a command to this register, the other registers containing information to be used during execution must be loaded and the busy bit (bit 7) of the status register must be cleared. Tables 16-30 and 16-31 provide coding information for this register. The commands are described in greater detail later in this chapter.

Table 16-30. Command Register Coding

COMMAND	BINARY CODE	HEX CODE
Test drive	0000 0000	00H
Cartridge change	00000000	00H
Restore	0001 XXXX	1XH (see note)
Read sector		
Data only with retry	0010 0000	20H
Data only, no retry	0010 000	121H
Data & ECC with retry	0010 0010	22H
Data & ECC, no retry	0010 0011	23H
Write sector		
Data only with retry	0011 0000	30H
Data only, no retry	0011 0001	31H
Data & ECC with retry	0011 0010	32H
Data & ECC, no retry	0011 0011	33H
Verify with retry	0100 0000	40H
Verify, no retry	0100 0001	41H
Format	0101 0000	50H
Seek	0111 XXXX	7XH (see note)
Diagnose	1001 0000	90H
Set parameters	1001 0001	91H

Mass Storage

Table 16-30 (continued). Command Register Coding

COMMAND	BINARY CODE	HEX CODE
Acknowledge cartridge change	1011 0000	B0H
Set drive type	1111 0000	F0H
Recovery mode on	1111 0001	F1H
Recovery mode off	1111 0010	F2H

NOTE: The low-order bits of this command byte set the step rate for the seek or restore operation. See Table 16-31 for specific programming information.

Both the seek and restore commands may be executed at step rates between 5 microseconds and 7.5 milliseconds. The step rate is programmed with the lower 4 bits of the command byte. Table 16-31 provides coding information for the variable step rates.

Table 16-31. Seek and Restore Step Rate Codes

STEP RATE	BINARY CODE	HEX CODE
5 μ s	1110	EH
10 μ s	0111	7H
20 μ s	1000	8H
30 μ s	1001	9H
35 μ s	0000	0H
40 μ s	1010	AH
50 μ s	1011	BH
60 μ s	1100	CH
70 μ s	1101	DH
0.5 ms	0001	1H
1.0 ms	0010	2H
1.5 ms	0011	3H
2.0 ms	0100	4H
2.5 ms	0101	5H
3.0 ms (see note)	0110	6H
7.5 ms	1111	FH

NOTE: Following execution of a reset or diagnose command, the controller sets the step rate to 3.0 milliseconds.

Fixed Disk Register (Port 3F6H)

This 8-bit write-only register enables the fixed disk interrupt and resets the fixed disk functions. It does not support reduced write current functions. Table 16-32 provides coding information for this register.

Table 16-32. Fixed Disk Register Codes

BIT	FUNCTION	DESCRIPTION
0	Not used	
1	Interrupt	When this bit is cleared, the fixed disk interrupt is enabled. Setting this bit disables the fixed disk interrupt.
2	Reset	When this bit is set, the fixed disk functions are reset.
3	Reserved	This controller does not support the reduced write current function.
4-7	Not used	

Digital Input Register (Port 3F7H)

This 8-bit read-only register reflects the control signals being sent to the drive during a read operation. Bits 0 through 6 reflect the status of the hard disk drive. Bit 7 indicates the status of the floppy disk changed signal. Table 16-33 provides decoding information for the digital input register.

Table 16-33. Digital Input Register Codes

BIT	FUNCTION	DESCRIPTION
0	Drive 0	This bit reflects the state of the drive 0 select signal sent from the controller.
1	Drive 1	This bit reflects the state of the drive 1 select signal sent from the controller.

Mass Storage

Table 16-33 (continued). Digital Input Register Codes

BIT	FUNCTION	DESCRIPTION
2	Head 0	This bit reflects the state of the head 0 select signal sent from the controller.
3	Head 1	This bit reflects the state of the head 1 select signal sent from the controller.
4	Head 2	This bit reflects the state of the head 2 select signal sent from the controller.
5	Head 3	This bit reflects the state of the head 3 select signal sent from the controller.
6	Write gate	This bit reflects the state of the write gate signal sent from the controller.
7	Floppy	This bit reflects the state of the floppy disk changed signal sent from the floppy disk drive.

Programming Hard Disk Operations

Programming hard disk operations in this computer is not simple since a variety of hard disk drives are accommodated. Be sure you are thoroughly familiar with the parameters for the drive you intend to program. As with floppy disk operations, an error in programming results in lost data at best. The software interrupts discussed in Chapter 10 contain all the routines necessary to carry out hard disk operations. Use these whenever possible to prevent lost data and drive damage. Also, refer to Programmer's Utility Package for additional information.

Hard Disk Controller Commands

The hard disk controller (HDC) accepts thirteen commands. In general, commands are executed in the following manner:

1. Output the drive, head, disk, and operation information to the appropriate hard disk control register ports.

2. Read the controller status register to verify that the controller is not busy, the write fault signal is inactive, and the drive ready and seek complete signals are active.
3. Output the command byte to the command register.
4. Read the controller status register and error register ports to verify that the command was properly executed.

The remainder of this chapter provides information about each of these commands.

Set parameters, set drive type — A maximum of two hard disk drives may be installed in this computer. They need not be the same drive type or have the same drive parameters. The HDC supports several different hard disk drives. The set parameters and set drive type commands notify the controller which drives have been installed in a particular system. These commands must be executed prior to attempting any multi-sector operations and after each power-up and reset. The drives are defaulted to fixed disk types at power-up and reset. The set parameters command is issued first, followed by the set drive type command. This command set must be issued for each hard disk drive installed in the system.

To execute the set parameters command, refer to Table 16-28 and output the drive number and the maximum number of heads in the drive to the head and drive register (port 1F6H) and the bit pattern 0001 0001 (11H) to the sector count register (port 1F2H). Then output the bit pattern 1001 0001 (91H) to the command register (port 1F7H).

To execute the set drive type command, output the drive number to the head and drive register (port 1F6H). If the drive is a fixed disk type, output the bit pattern 0000 0000 (00H) to the cylinder high register (port 1F5H). If the drive is a removable cartridge type, output the bit pattern 0000 0001 (01H) to the cylinder high register (port 1F5H). Then output the bit pattern 1111 0000 (F0H) to the command register (port 1F7H).

Mass Storage

Test drive/cartridge change — This command tests the state of the drive ready and cartridge changed signals from the drive. It is executed by outputting the drive and head select bit pattern to the drive and head register (port 1F6H) and then the bit pattern 00000000 (00H) to the command register port (port 1F7). Refer to Table 16-28 for the drive and head register codes.

Acknowledge cartridge changed — This command resets the cartridge changed signal from a removable cartridge drive by deselecting and then reselecting the drive. If the controller is in hardware drive select mode, the drive to be reset must be deselected and reselected through the head and drive register before this command can be executed. Refer to Table 16-28 for drive and head register codes. This command must be issued for each drive indicating a cartridge changed error.

To execute this command, deselect and reselect the drive indicating an error by outputting the head and drive code bytes to the head and drive register (port 1F6H), and then output the bit pattern 1011 0000 (B0H) to the command register (port 1F7H).

Restore — For fixed disk drives, this command sends a series of 2047 step pulses to the selected drive. After each pulse, the controller monitors the seek complete and track 00 signals from the drive. If track 00 has not been found after all 2047 steps have been executed, the controller sets the error bit in the status register. Upon completion of the command, an interrupt is generated.

For removable cartridge drives, this command sends a burst of 2047 pulses to the selected drive. After receiving a seek complete signal, the controller checks the state of the track 00 signal. Again, an interrupt is generated upon completion of the command.

This command is executed by outputting the appropriate drive and head select code to the drive and head register (port 1F6H) and then outputting the bit pattern 0001 and the 4-bit step rate code to the command register (port 1F7H). Refer to Table 16-31 for the step rate code and Table 16-28 for the drive and head register code.

Format track — This command formats the track specified by the drive and head, cylinder high, and cylinder low registers. Sector interleave is programmable, allowing up to 17-way interleave. A sector interleave table for 17 sectors followed by a pad of 478 bytes of 00H must be output to the data register for each track to be formatted. The controller ignores the pad and issues an interrupt upon completion of the command. The format track command may be repeated without issuing a restore command, provided the tracks are all located on the same disk. A restore command must be issued when switching disks.

The controller allows a 17 sector per track format. It will not operate properly if another format is attempted. The sector layout for this format, described in Table 16-34, is repeated 17 times on each track.

Table 16-34. Format Track Command Track Layout

FUNCTION	BYTES	VALUE	DESCRIPTION
SPEED GAP	30	4EH	These bytes provide the speed-tolerance gap. Their presence assures a 4.19% speed tolerance.
SYNC1	14	00H	These bytes provide a synchronization signal for the phase locked loop (PLL).
IDAM	1	A1H	This is the first byte of the address ID mark.
IDEN	1	XXH	This is the last byte of the address ID mark.
CYL	1	XXH	This byte contains the current cylinder number.
HEAD	1	2XH	This byte contains the current head number.
SECTOR	1	XXH	This byte contains the logical sector number.

Mass Storage

Table 16-34 (continued). Format Track Command Track Layout

FUNCTION	BYTES	VALUE	DESCRIPTION
CRC	2	XXH	These bytes contain the cyclical redundancy code for the ID field.
SYNC 2	15	00H	The first three bytes provide write splice. The write gate should open immediately after the third byte of this field. The remaining 12 bytes provide PLL sync.
DAM	1	A1H	This is the first byte of the data address mark.
DIN	1	F8H	This is the last byte of the data address mark.
DATA	512	XXH	These bytes are the data storage area. A maximum of 256 16-bit words are stored high byte first.
ECC	4	XXH	These are the error correction code bytes.
SYNC 3	3	00H	These bytes provide the write turn off area.

The sector interleave table is comprised of two bytes for each sector. The first byte indicates whether the sector is a good or a bad block. The second byte indicates the logical sector located in the particular physical sector. Table 16-35 gives an example of a 2 to 1 sector interleave table. The example indicates that all the sectors in the track are good blocks. Entering an 80 in place of the 00 byte indicates a bad block. A physical sector number is provided in the table to help you understand the sector layout on the track. Only the logical sector number, the sector indicator, and the pad of 00H bytes are entered into the data register.

Table 16-35. 2 to 1 Sector Interleave

PHYSICAL SECTOR NUMBER	SECTOR INDICATOR	LOGICAL SECTOR NUMBER
01H	00H	01H
02H	00H	0AH
03H	00H	02H
04H	00H	0BH
05H	00H	03H
06H	00H	0CH
07H	00H	04H
08H	00H	0DH
09H	00H	05H
0AH	00H	0EH
0BH	00H	06H
0CH	00H	0FH
0DH	00H	07H
0EH	00H	10H
0FH	00H	08H
10H	00H	11H
11H	00H	09H

Seek — This command causes the selected drive to seek to the cylinder indicated by the cylinder high and cylinder low registers. The controller does not wait for a seek complete signal before issuing a task complete interrupt. The head position is not verified.

To execute this command, output the drive and head code to the head and drive register (port 1F6H), output the selected cylinder number to the cylinder high (port 1F5H) and cylinder low (port 1F4H) registers, and output the bit pattern 0111 and the step rate code to the command register. Refer to Table 16-31 for the step rate code and Table 16-28 for the drive and head register code.

Read sector — This command causes the controller to automatically seek to the head, cylinder, and sector indicated by the drive and head register, cylinder high and cylinder low registers, and sector number register and read from 1 to 256 sectors as indicated in the sector count register. An interrupt is sent to the microprocessor upon completion of each sector, notifying it to read the data register. No interrupt is generated at the end of the command.

Mass Storage

The command can be executed in either data and ECC mode or in data only mode. If the command is executed in data and ECC mode, sectors of 516 bytes are transferred. In this mode, a maximum of five data byte errors may be corrected without terminating the command. If the command is executed in data only mode, sectors of 512 bytes are transferred.

If the command is executed in retry mode, up to 16 attempts will be made before the command is terminated. If retry mode is disabled, two attempts are made before the command is terminated. The command is executed at the same step rate as specified in the last seek or restore command. The command is only executed following a successful match of the sector ID with the targeted ID.

To execute this command, output the drive and head information to the drive and head register (port 1F6H) using the information from Table 16-28, and output the logical starting cylinder number in hex to the cylinder high (port 1F5H) and cylinder low (port 1F4H). Next, output the number of sectors to be read in hex to the sector count register (port 1F2H), output the logical starting sector number in hex to the sector number register (port 1F3), and output the read command bit pattern to the command register (port 1F7H) using the information from Table 16-30. Finally, input the data from the data register (port 1F0H).

Verify — This command is used to verify the integrity of a hard disk. It is similar to the read command but sends no data to the microprocessor. The controller seeks to the drive, head, cylinder, and sector indicated by the head and drive register, the cylinder high and cylinder low registers, and sector number register, reads the data and ECC bytes for the number of sectors specified in the sector count register, checks the ECC bytes, and generates an interrupt when an error is encountered or when the command is completed.

Write sector — This command causes the controller to automatically seek to the drive, head, cylinder, and sector indicated by the drive and head register, cylinder high and cylinder low registers, and sector number register and write the number of sectors indicated by the sector count register. After the first sector has been transferred, an interrupt is sent to the microprocessor before each subsequent sector is transferred. Another interrupt is generated upon completion of the command.

The write sector command can be executed in either data and ECC mode or in data only mode. In data only mode, 512 bytes per sector are transferred. In data and ECC mode, 516 bytes per sector are transferred. In this mode, a maximum of five data byte errors may be corrected automatically without terminating the command.

If this command is executed in retry mode, up to 16 attempts will be made before the command is terminated. If retry mode is disabled, two attempts are made before the command is terminated. The command is executed at the step rate specified in the last seek or restore command. The command is only executed following a successful match of the sector ID with the targeted ID.

To execute this command, output the drive and head information to the drive and head register (port 1F6H) using the information from Table 16-28. Next, output the number of cylinders to be written to the sector count register (port 1F2H), output the logical starting sector number to the sector number register (port 1F3H), the starting cylinder number to the cylinder high (port 1F5H) and cylinder low (port 1F4H) registers. Finally, output the write sector command bit pattern to the command register (port 1F7H) using the information from Table 16-30 and the data to the data register (port 1F0H).

Diagnose — This command initiates the controller's self-tests. The results of the tests are reported in the error register. Upon completion of the tests, an interrupt is generated. To execute this command, output the bit pattern 1001 0000 (90H) to the command register (port 1F7H) and input the result from the error register (port 1F1H).

Set recovery mode on, set recovery mode off — These commands assert and deassert the recovery line to a removable cartridge drive. To operate in recovery mode, the recovery mode on command must be issued at power-up and after each reset, since the controller deasserts the recovery line at power-up and on reset. The commands must be issued for each removable cartridge drive. They are used only with removable cartridge drives and result in an aborted command error if used with a fixed drive.

Mass Storage

To execute the set recovery mode on command, refer to Table 16-28 and output the drive select code to the drive and head register (port 1F6H). Then output the bit pattern 1111 0001 (F1H) to the command register (port 1F7H).

To execute the set recovery mode off command, refer to Table 16-28 and output the drive select code to the drive and head register (1F6H). Then output the bit pattern 1111 0010 (F2H) to the command register (port 1F7H).

Chapter 17

31 kHz Video

This chapter provides an overview of various types of video systems. Basic video system technology and CRT controller concepts are discussed and a description of basic video modes is provided. The next section of the chapter gives a very brief description of specific video formats currently accepted.

The section that follows the video systems overview provides specific information about the 31 kHz video card. Each emulated video system is described with specific register programming information. The last section in this chapter provides operating system support for the 31 kHz video system.

Video Systems Overview

This section of the manual provides basic information about some of the types of video systems that are currently used. The discussions address raster-scan technology, the most popular technology used with personal computers, the way that a typical CRT controller device works, and various types of video standards available today. These discussions are general in nature and do not reflect the specific configuration of the computer this manual supports. Specific information about the video interface for this computer is provided in the next section.

Raster-Scan Technology

Each type of video interface must be capable of displaying information on some type of video monitor. This section describes the display technique used with most current video interfaces.

31 kHz Video

Two different display types are considered raster-scan devices; LCD and external CRT monitors. A raster is a predetermined pattern of lines that are scanned to create the image on the screen. Raster-scan displays are most commonly used in microcomputer applications. Although the two operate differently (the CRT produces light while the LCD blocks light), they both produce recognizable images by activating individual points called pixels (picture elements) as the display is scanned.

The scanning of the display takes place in a left-to-right, top-to-bottom motion. When the scanning reaches the end of a line, it moves down one line and starts over at the left side of the screen. When the scanning reaches the bottom of the screen, it returns to the upper-left corner and the process starts over. Figure 17-1 illustrates the timing relationships of the signal that produces the display you see on the screen. The relationships shown are not absolute, but only roughly represent the actual signals.

The screen of a typical PC display is divided into a matrix of pixels 200 rows deep by 640 columns wide. Each pixel forms part of the display. A character (letter, number, or punctuation mark) is formed in a matrix (character cell) producing a display of 80 characters wide by 25 characters deep (depending on the actual video mode).

In the figure, which is divided into three sections, the video signal itself is not illustrated. The top section illustrates the timing that makes up one raster-scan line. During one character clock cycle, the video signal transmits eight pixel locations to the screen. This continues until the number of character clock cycles equals the number of characters to be displayed in one horizontal line. The character clock continues to cycle while the beam in the CRT is returned to the left side of the screen (this is called the horizontal retrace period). From that point the character clock continues to cycle until the total number of horizontal character positions has been reached, from which point the process starts over with the

next row of pixels sent to the screen. The video controller contains registers that hold the number of horizontal characters to be displayed; the time, in characters for horizontal retrace; and the total number of character positions in one line.

The second section of the figure illustrates the timing needed for one character line. The character clock and position are illustrated on the first two lines of this section. The third line, the raster line, illustrates the time required for one raster-scan line, as illustrated in the first section of this figure. The raster lines continue to be scanned until the maximum raster address for a character, which is stored in a register of the video controller, is reached. This value is usually eight, representing the eight rows of pixels that make up a character frame. Since this value can be reprogrammed, it is possible to display characters that contain more than eight rows of pixels. For instance, video mode 7 supports characters that are nine pixels wide and 14 rows deep.

The third section of the figure illustrates the relationship between one character line and the entire screen, called a frame. In this section, the top line illustrates the characters line, one of which is illustrated in the second section of this figure. After a predetermined number of lines have been displayed, additional blank lines are sent to the display to fill the screen, both at the bottom and the top. During this time, the scan returns to the left and top of the screen (vertical retrace).

Since each pixel may be selectively activated, individual alphanumeric and graphics characters or images that take part or all of the display's area are produced. The video memory of the computer, which determines whether pixels are activated or not, is addressed in the same manner that scanning takes place: in a left-to-right, top-to-bottom pattern, whether an LCD or CRT display is used.

31 kHz Video

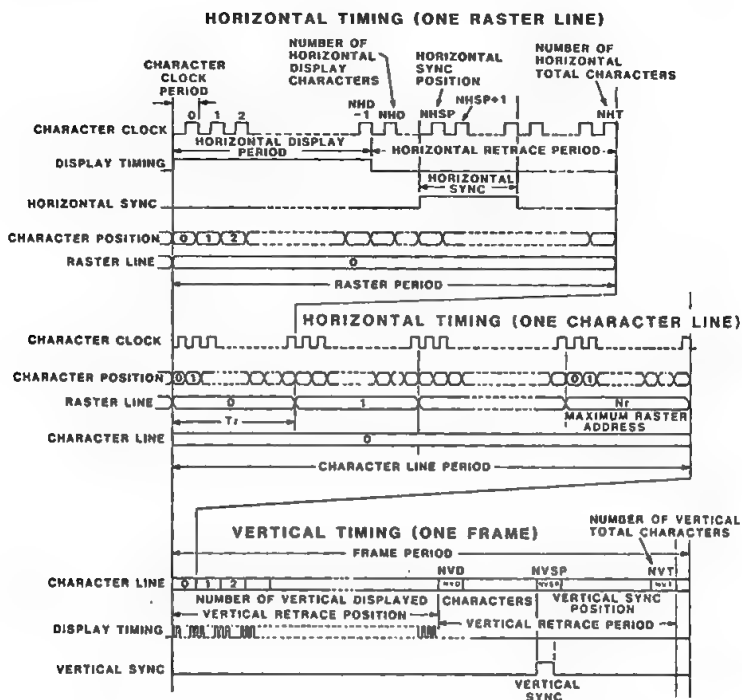


Figure 17-1. Display Signal Relationships

The scanning of a typical CRT is illustrated in Figure 17-2. The display area is that area of the CRT where data is displayed and is 640 pixels wide by 200 pixels high. Electronic circuits in the monitor supply the necessary operating voltages to fire a beam of electrons at phosphors that are deposited on the inside face of the CRT. The beam, as it strikes the phosphors, agitates or excites them, producing visible light for each pixel. The color of the phosphor determines the color of the pixel. In monochrome monitors there is only one type of phosphor and only one electron beam. The intensity of the beam determines the amount of light given off by each pixel, giving the monitor the ability to produce different shades of light, called a gray scale.

In color monitors three beams are fired at three differently colored phosphors that are deposited on the inside face of the CRT, producing red, green, and blue light. By firing one, two, or three beams

at the different phosphors at any one pixel location, and in varying intensities, the monitor can display the different colors you see on a color CRT.

In the illustration, the border area is that area outside the display area. Since the phosphor coating does not always extend to the edge of the tube, any areas not coated with phosphors are not illuminated by the electron beam.

However, scanning the CRT actually exceeds the area coated by phosphors, and often exceeds the width and height of the CRT. This overscan area, which is shown in the illustration, allows the full screen of the CRT to be used for display purposes. The beginning and end of the synchronization pulses, illustrated in Figure 17-2 as horizontal sync (in the first section) and vertical sync (in the third section), cannot be observed, because this takes place outside the overscan area. These pulses establish signal blanking, wherein the beam of electrons is either turned off or reduced to the point where it no longer excites the phosphors on the face of the CRT.

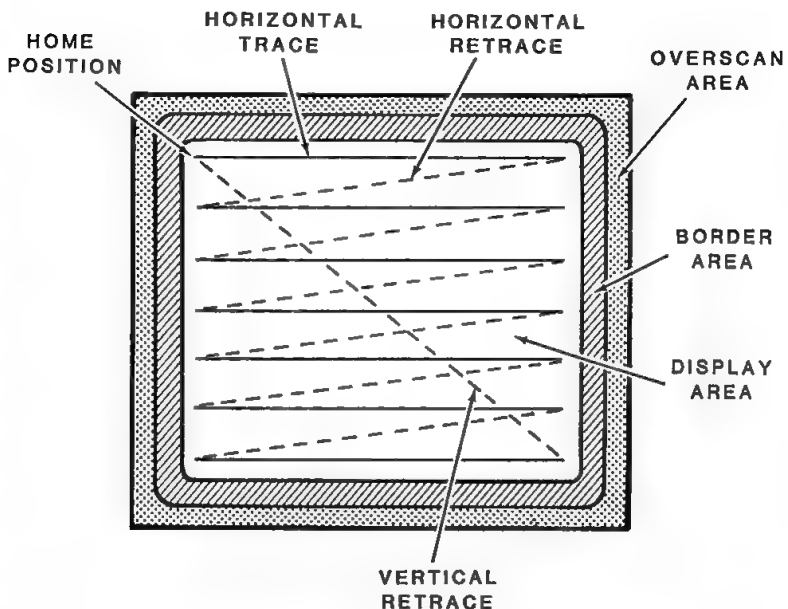


Figure 17-2. Scanning the Face of the CRT

31 kHz Video

Liquid crystal displays do not emit light. Instead, they depend upon two ways to produce a readable display: reflected light or light sent through the LCD. The majority of clocks, including watches, that have LCDs use room light that is reflected through the LCD to produce a display. Even at night, when you press the buttons on your watch to light its display, the light is reflected through the LCD. Some newer LCD displays, send light directly through the LCD. By selectively blocking this light, whether it is reflected or transmitted, an image can be produced on the LCD.

One of the materials used in liquid crystal displays is held between a liquid and crystalline (solid) state through a wide temperature range ($32 - 140^{\circ}\text{F}$ or $0 - 60^{\circ}\text{C}$). When the material is electrically charged, the electrons in the material align themselves so light can pass through the material. When they are not aligned, or polarized, the material is opaque and light cannot pass through it. By allowing light to pass through the material while in a transparent state (charged), the opaque (not charged) areas become visible to the viewer. High contrast and a wide viewing angle is obtained by using high polarization angles and a very fine grid. An optional backlight makes it possible to use the display under all lighting conditions.

In most LCD monitors, the liquid crystal material is encased between two transparent plates. A conductive coating is etched on the inside of one of the plates in the form of a matrix 640 columns wide by 200 rows deep. This matrix is driven by row and column drivers that can be turned on to charge the liquid crystal material and make it transparent, lighting a pixel, or turned off, leaving the pixel opaque. The drivers are controlled by the video controller that addresses (scans) the drivers by column and row, starting at the uppermost and leftmost point on the screen, moving to the right until the end of the row is reached, returning to the beginning of the next row, and so on, until the entire screen has been "scanned." The alignment of the material for individual pixels can be precisely controlled allowing the pixel to be partially turned on, producing a gray scale display.

CRT Controller Concepts

The video controller is the interface between the computer system and the video monitor. It receives information from the computer system, processes it into a form that can be displayed on the video monitor, and then sends the information, along with the proper operation (sync) signals, to the monitor.

The video controller is the key active element that determines the video performance of a workstation. No matter what the video display's potential, without the proper signals from the controller, the display will not perform to its potential. In comparison to a component stereo system, the display is analogous to the speakers, while the controller is analogous to the amplifier's output circuits.

In order to accomplish their jobs, video controllers must be able to process enormous amounts of information in a timely manner. Consider that even a low-resolution system, such as a microcomputer-based workstation with a resolution of 640×225 has 144,000 separate pixels. It is the video controller's job to monitor and report to the video display the location and status of each pixel. It must do this 60 times a second.

Consider a workstation with a high-resolution display such as a $1,024 \times 1,024$ display. The controller in this system must furnish information about over 1,000,000 pixels during each raster scan. The video controller ensures that the information about each of these pixels is accurate and in the proper sequence.

The flow of information through a video controller is straight forward. Binary information from the central processor is stored in the video memory. These memories communicate with the shift registers which quickly move the information to the encoder. The encoder processes video information from the memories and sync signals from the controller IC. The encoder then sends the signals on to the video display.

31 kHz Video

Controller IC

In many early computer systems, the video display was controlled by the central processing unit itself. However, as screen resolution increased, controlling the video display took so much central processor time that the entire system slowed dramatically. The solution was to add intelligence to the video controller.

Added intelligence most often takes the form of a special microprocessor called a video controller (or CRT controller) chip. These microprocessors are specialized devices that have been optimized to control video information.

Adding intelligence to the video controller accomplishes two things: one, it optimizes flow of video information to the video display, and two, it frees the central processing unit to do more work. This enables both types of processors to work at the same time.

But the controller IC does not work alone. It must work with the other components in the system. One component that both the central processor and the video controller IC use is the video memory.

Video Memory

Video memory stores binary information that consists of two logic states, 0 and 1. Each memory location corresponds to a particular screen pixel which can be either on or off. The entire video memory area is sometimes called a bit map, or a video plane.

A monochrome monitor only turns pixels on or off; hence, it only requires a single video plane. Because a color monitor needs more information, such as the color of the pixel, as well as its on/off status, color controller conventionally use several planes of video memory.

Some video controllers are designed so that more video planes can be added later. Increasing the number of video planes allows the system to display more colors. For example, the addition of another memory plane to control the intensity of the electron beam to the color controller enables a system to display 16 different colors.

The screen memory is normally constructed from dynamic RAM integrated circuits. Dynamic means that these chips must be periodically refreshed to prevent them from losing data. But while the memory is being refreshed, it cannot be accessed for screen display.

If the memory needs to be refreshed at the same time that it needs to supply the video display with information, a conflict will occur. To prevent this kind of conflict, video controllers use shift registers to supply information to the video display.

Shift Registers and the Video Encoder

Shift registers make sure that video information is available in the proper form when it is needed on the screen. These are specialized devices that accept binary information (in parallel) from the video memory and send a serial data stream to the video encoder..

Before the video information can be used by the video display, it must be combined with the sync signals that the display requires for proper operation. The video encoder accepts information from the shift registers and the controller IC, combines them in the proper sequence and sends the signals to the video display.

Hardware Features

There are several hardware features that are dependent upon the video controller hardware. As an example, some controller have built in circuits that can automatically execute commands. Some computer-aided design packages take advantage of these special features to dramatically reduce the time required to execute certain functions.

The amount and speed of the video memory limits the video capabilities of most systems. In order to maximize flexibility, some systems are designed to be expandable. These systems allow you to increase the video performance of the system as your needs require.

31 kHz Video

An alternate video controller architecture uses a color encoder rather than separate video planes. In operation, the video memory does not directly control the electron beam but holds various color codes which are fed to a color encoder, which in turn drives the electron beam.

Some systems can use video memory in ways other than those previously described. For example, rather than dividing the memory into video planes, it can be divided into video pages.

When the memory is divided into video pages, one page can be used to control the display while the other page is updated. Then the pages can be flashed one after another on the video display to effect kinematic simulations or even animation.

Other controllers use multiple video controller chips in parallel. In this situation, each video plane is assigned its own controller chip. With the proper supporting components, this can increase the performance of a system dramatically.

Basic Video Modes

Video memory consists of separately addressable, but parallel, 64K memory planes. Depending upon the video mode selected, this memory may be combined in different ways. The following discussion describes the basic modes of operation and how they work with the video memory.

Text (Alphanumeric) Modes — A two-byte character/attribute format is used to define each character display position on the screen. These two bytes are mapped into assigned locations in video memory with the character byte in memory plane 0, and the attribute byte in memory plane 1.

Normally, bit 3 of the attribute byte is used by display cards to determine the intensity characteristic of the displayed character. However, some expanded video cards allow bit 3 to be redefined by the character map select register. The bit is then used as a switch to move between character sets. This gives the programmer access to both character sets, for a total of 512 characters, instead of just one character set. To accommodate the additional 256 characters, the patterns of both character generators are transferred into memory plane 2 of the video memory. In addition, with 256K of video memory on this card, it is possible to support two additional user-defined 256-member character sets for a total of 1024 different characters. The patterns of these additional character sets must also be loaded into memory plane 2 of the video memory.

To display a screen of data, the CRT controller generates a series of sequential addresses, which represents the sequential locations of characters on the screen. These addresses correspond to video memory addresses where memory planes 0 and 1 are accessed. The byte in memory plane 0, combined with the scan row count, determines the address of the bit pattern stored in memory plane 2. If bit 3 of the attribute byte has been selected to as a switch between character sets, the video memory address is modified accordingly. The bit pattern found in memory plane 2 is then assigned the attributes found in the attribute byte and sent as a serial stream of data out the appropriate pins of the video connector.

Monochrome and color attribute data is formatted differently. Table 17-1 describes the attribute byte characteristics for monochrome modes.

31 kHz Video

Table 17-1. Monochrome Attribute Byte Characteristics

7	6	5	4	BIT			0	ATTRIBUTE DESCRIPTION
				3	2	1		
B	0	0	0	1	1	1	1	Normal video
B	1	1	1	1	0	0	0	Reverse video
B	0	0	0	1	0	0	0	No display (black output)
B	1	1	1	1	1	1	1	No display (white output)

NOTE: Bit 7 defines whether the character blinks on and off. If bit 7 is a 1, the character will blink. If bit 7 is 0, the character will not blink. Bit 3 defines either the intensity characteristic or the character set. Character set definition is determined by the character map select register. If bit 3 controls the intensity attribute, then a 1 will cause the character to be displayed at high intensity; a 0 will cause the character to be displayed at normal intensity.

Table 17-2 describes the attribute byte characteristics for color modes. The bit pattern for the foreground color and background color are identical. The foreground color attribute occupies bits 4-6.

Table 17-2. Color Attribute Byte Characteristics

BACKGROUND BIT			FOREGROUND BIT			NORMAL COLOR	INTENSIFIED COLOR
6	5	4	2	1	0		
0	0	0	0	0	0	Black	Dark Gray
0	0	1	0	0	1	Blue	Light Blue
0	1	0	0	1	0	Green	Light Green
0	1	1	0	1	1	Cyan	Light Cyan
1	0	0	1	0	0	Red	Light Red (Pink)
1	0	1	1	0	1	Magenta	Light Magenta
1	1	0	1	1	0	Brown	Yellow
1	1	1	1	1	1	White	Intensified White

NOTE: Bit 7 defines whether the character blinks on and off. If bit 7 is a 1, the character will blink. If bit 7 is a 0, the character will not blink. Bit 3 defines either the intensity characteristic or the character set. Character set definition is determined by the character map select register. If bit 3 controls the intensity attribute, then a 1 will cause the character to be displayed at high intensity; a 0 will cause the character to be displayed at normal intensity.

Graphics Modes — With the exception of video mode F (high-resolution monochrome) and video mode 10 (enhanced color graphics), basic video mode addressing, mapping, and data formatting are the same as those used with Z-100 PC color video cards and PC-equivalent color/graphics adapters.

In the Z-100 PC color video graphics modes, two degrees of resolution are available: 320×200 and 640×200 . 640×200 is obtainable only as a monochrome, or two-color, display (one of 16 border colors may be used for the displayed pixels). In the 320×200 resolution modes, each displayed pixel may be one of four colors selected from two palettes and the border color.

In the medium resolution color mode pixel information for the 320×200 display is stored in two memory planes of 8000 bytes each, starting at address B8000H. The even-numbered (0, 2, 4, ... 198) scan row information is stored from B8000H to B9F3FH. The odd-numbered (1, 3, 5, ... 199) scan row information is stored from BA000H to BBF3FH.

Each byte of video memory contains four two-bit pairs, which define the color displayed for the pixel. Note that there is no off or unlit pixel capabilities: therefore, if you want to be able to unlight any given pixel, you will have to sacrifice one of the possible colors (the background color) to black.

Table 17-3 defines the color selected for the defined pixel. Since only one palette can be active, Table 17-4 defines the colors in the two possible palettes for these video modes.

Table 17-3. Color Selection

MSB	LSB	COLOR DEFINITION
0	0	The pixel will display the current background color.
0	1	The pixel will display color 1 of the current palette.
1	0	The pixel will display color 2 of the current palette.
1	1	The pixel will display color 3 of the current palette.

31 kHz Video

Table 17-4. Palette Colors

COLOR NUMBER	PALETTE 1	PALETTE 2
1	Cyan	Green
2	Magenta	Red
3	White	Brown

The 16 border colors available are the same that can be used to display text characters: Black, Blue Green, Cyan, Red, Magenta, Brown, White, Dark Gray, Light Blue, Light Green, Light Cyan, Light Red(Pink), Light Magenta, Yellow, and Intensified White.

In the high resolution color mode the entire memory complement in a Z-100 PC color video card (16K) is needed to define the on or off state of a full screen of pixels. Each pixel location on the 640 × 200 screen is represented by a location in video memory, starting at B8000H. Because each location can be set only a 1 or a 0, each pixel may be the border color or not lit. The 16 border colors are: Black, Blue, Green, Cyan, Red, Magenta, Brown, White, Dark Gray, Light Blue, Light Green, Light Cyan, Light Red (Pink), Light Magenta, Yellow, and Intensified White.

The high-resolution 640 × 350 mode (mode F) supports monochrome black and lit-pixel graphics with the following attributes: black, video, blinking video, and intensified video. Two memory planes, required to support the four attributes, are both addressed at A0000H by the CPU. The first memory plane is identified as the video memory plane and the second memory plane is identified as the intensity memory plane. Actually, a combination of pixel pairs between the two memory planes is required to define the attributes as shown in Table 17-5.

Table 17-5. Mode F Attributes

ATTRIBUTE	VIDEO MEMORY PLANE	INTENSITY MEMORY PLANE
Black(not lit)	0	0
Video(lit)	0	1
Blinking video	1	0
Intensified video	1	1

NOTE: Some IBM-equivalent Enhanced Graphics Adapters are supplied with only 64K of video memory. When this is the case, memory planes 0 and 1 and memory planes 2 and 3 are mapped together to form the two 32K memory planes needed to support this mode. Although the four memory planes all reside at an A0000H starting address, memory planes 0 and 2 are addressed as even memory planes and memory planes 1 and 3 are addressed as odd memory planes. When data is brought out of video memory for display, the first byte comes out of the even memory planes, the second out of the odd memory planes, the third out of the even, and so on.

Other EGA cards have 256K of memory, so the memory planes are not mapped together as described in the note. The first (memory plane 0) and third (memory plane 2) are used for mode F. The second (memory plane 1) and fourth (memory plane 3) are ignored.

Two bits, one from each plane, define each pixel on the screen. Since memory organization is sequential in nature, the first eight pixels are contained in the first byte, starting at video memory address A0000H and with the most significant bit. The second byte is located at video memory address A0001H and so on.

Mode 10 supports graphics in 16 colors, selected from 64 possibilities. All four available memory planes in video memory are used. Each memory plane normally represents a color or intensity, as shown in Table 17-6. The base colors that can be produced without programming additional registers are described in Table 17-7. The other possible shades are obtained by programming the palette registers in the attribute controller.

Table 17-6. Mode 10 Memory Plane Assignments

MEMORY PLANE	COLOR OR FUNCTION
0	Blue
1	Green
2	Red
3	Intensified

31 kHz Video

Table 17-7. Mode 10 Base Colors

BP3	BP2	BP1	BP0	RESULTING COLOR
0	0	0	0	Black
0	0	0	1	Blue
0	0	1	0	Green
0	0	1	1	Cyan
0	1	0	0	Red
0	1	0	1	Magenta
0	1	1	0	Brown
0	1	1	1	White
1	0	0	0	Dark Gray
1	0	0	1	Light Blue
1	0	1	0	Light Green
1	0	1	1	Light Cyan
1	1	0	0	Light Red
1	1	0	1	Light Magenta
1	1	1	0	Yellow
1	1	1	1	Intensified White

There are 16 palette registers that correspond to the 16 possible base colors. The registers allow the user to remap the color represented by the bit patterns shown in the preceding table to any one of 64 possible colors that can result from combining the six bits in each register. Refer to Table 17-8 for the bit-to-color relationship. Setting a bit to 1 activates that color. Setting the bit to 0 deactivates that color.

Table 17-8. Palette Register Color Attributes

BIT	COLOR
0	Blue
1	Green
2	Red
3	Secondary blue
4	Secondary green
5	Secondary red

Video Systems

There are many video systems that provide varying degrees of resolution and graphics capabilities. The following sections give a very brief description of the most common systems available today. These systems are considered to represent video standards. Other video interfaces capable of even greater resolutions are produced but this manual does not address them.

MDA System

Many different video interface systems are currently available for most computer designs. Historically, the development of the video interface began with IBM's MDA (monochrome display adapter). The MDA interface supports 4K of video memory and is only capable of displaying alphanumeric characters and some special characters. The MDA interface is not capable of generating graphic displays.

The advantage of the MDA interface is its high quality of character resolution. A 9-pixel by 14-pixel character cell contains the character that is displayed. The actual character is composed of a 9-pixel by 7-pixel format. The remaining pixels in the character cell can provide video attributes such as blinking, underlining, and reverse-video imagery. The MDA interface is generally used with a high-resolution monochrome video monitor.

HGC System

Shortly after the MDA video interface became popular, Hercules Computer Technology introduced an enhanced monochrome display capable of generating graphics. The Hercules Graphics Card (HGC) contains 64K of video memory and provides 720-pixel by 348-pixel graphic resolution. The card maintains the character resolution and display size of the MDA standard but also supports 132-column by 25-row alphanumeric and 132-column by 44-row alphanumeric modes of operation.

31 kHz Video

The HGC is widely used because of its extremely sharp resolution and monochrome graphics capability. In addition, various software development vendors are specifically supporting the HGC in their applications packages. Lotus 1-2-3 is a popular program that is designed to support the HGC. As a result, the HGC is fast becoming a standard in the video interface market.

CGA System

As technology improved and demand for graphics-intensive applications grew, the CGA (color graphics adapter) video interface was developed. This video interface is supplied with 16K of video memory and supports various text and graphics modes of operation. An 8-pixel by 8-pixel character cell contains the character that is displayed. The actual character is composed of a 7-pixel by 7-pixel format. CGA resolution is not as high as MDA resolution but the CGA interface is capable of generating medium-resolution graphics and up to four different colors from a palette of 16 possible colors.

EGA System

As the video interface became a more important aspect of a computer system, higher resolution and more capabilities were required. Specific applications such as CAD (computer-aided design), and desktop publishing brought forth the development of the EGA (enhanced graphics adapter) card. Supplied with 256K of video memory, the EGA card supports 12 distinct video modes. It is capable of high-resolution graphics (640-pixels by 350-pixels) and can display 16 colors out of a palette of 64 possible colors.

Some EGA cards support the monochrome modes provided by the MDA interface (with the same degree of resolution) and the color-graphics modes provided by the CGA interface. For the most part, the EGA card maintains CGA compatibility with programs that access the display through the system BIOS (basic input/output system). However, some programs that attempt to access the CGA video controller (6845) directly may not be compatible with the EGA card. The EGA card does not use a 6845 CRT controller. Most are designed with a custom chip set that emulates the 6845.

VGA System

The VGA (video graphics array) video interface operates very much like the EGA standard. However, some additional features have been added. For example, the VGA interface is capable of 640-pixel by 480-pixel resolution but displays square pixels on the screen. Standard EGA designs display rectangular pixels which can introduce distortion in some graphics applications. For example, if you display a circle with an EGA video interface (rectangular pixels), the aspect ratio of the circle can appear to be distorted and an oval might be displayed.

If the VGA video interface is run in its typical high-resolution 640-pixel by 350-pixel mode, it displays rectangular pixels like an EGA card. Square pixels are only available in the enhanced EGA mode (640-pixels by 480-pixels) of the VGA interface.

PGC System

At the high end of the spectrum is the PGC (professional graphics card) video interface. This interface is designed specifically for applications such as CAD. The card is supplied with 320K of video memory and supports 640-pixel by 480-pixel resolution. The PGC interface is capable of displaying 256 colors from a palette of 16 million possible colors. However, the PGC video interface is not fully compatible with previous designs.

31 kHz Video Card

The 31 kHz video card supports software designed for the MDA video interface, the HGC video interface, the CGA video interface, and the EGA video interface. The card's enhanced chip set emulates these four video formats while operating at a constant scan frequency of 31.49 kHz. It also supports an enhanced EGA mode that generates a 640 × 480-pixel resolution. Automatic switching circuits detect the type of video format required by software and place the card in the appropriate operating mode.

31 kHz Video

The following sections describe the 31 kHz video interface, the video modes that it emulates, and the enhanced mode that it supports. The discussion is segregated into three parts; 6845 CRT controller emulation, specific video mode emulation, and operating system support.

6845 CRT Controller Emulation

This section provides a bit-level analysis of the registers responsible for emulating the functions of the popular 6845 CRT controller IC. There are three video modes that are specifically handled by the emulated 6845 register set; MDA, HGC, and CGA.

An overall description of the emulated register set is given, with individual differences between video modes (MDA, HGC, and CGA) addressed in the following three sections. Note that the emulated 6845 register set (or some individual registers) are described with respect to the 6845 controller. Remember that the 31 kHz video card does not use a 6845 CRT controller IC but relies on other circuits to emulate 6845 functions.

The 6845 is a CRT controller that provides the signals necessary to produce a composite monochrome and RGBI video signals for use with raster-scan CRT video monitors.

Full compatibility with the IBM color graphics adapter (CGA) is maintained by the 6845 and its support circuits. Software written to CGA specifications will execute through the 6845 without need for modification.

The control section handles decoding of the control signals and communication between the internal register set and the data bus. The 6845 support circuits process the video memory data with information from the internal registers to produce video signals. The sync generator uses register information to produce the separate sync signals and combine them for the monochrome video signal.

The port addresses used by the 6845 video controller are identified in Table 17-9.

Table 17-9. 6845 Input/Output Port Addresses

ADDRESS	DESCRIPTION
3D0H	6845 pointer address port
3D1H	6845 data port
3D2H	6845 pointer address port
3D3H	6845 data port
3D4H	6845 pointer address port
3D5H	6845 data port
3D6H	6845 pointer address port
3D7H	6845 data port
3D8H	Video mode select port
3D9H	Color select port
3DAH	Status port
3DBH	Light pen clear port (not used)
3DCH	Light pen preset port

The 6845 ports are duplicated through "don't care" bits. Ports 3D0H, 3D2H, 3D4H, and 3D6H are all the same 6845 pointer address port. Likewise, 3D1H, 3D3H, 3D5H, and 3D7H are all the same 6845 data port.

Pointer Address Register

This write-only, 5-bit register serves as a pointer to the 18 6845 data registers. To address one of the 6845 data registers, you must first write the register number (00H-11H) into this register by executing an OUT instruction to one of the four 6845 pointer address ports (3D0H, 3D2H, 3D4H, or 3D6H). Then either read from (by executing an IN instruction) or write to (by executing an OUT instruction) to one of the four 6845 data ports (3D1H, 3D3H, 3D5H, or 3D7H).

Data Registers

The data port serves as the read/write port to the 6845 registers. These registers are described in Table 17.10. The initialization value for each register is described in Chapter 11.

Table 17-10. 6845 Data Registers

REGISTER	NUMBER	DESCRIPTION
R0	00H	Horizontal total register
R1	01H	Horizontal displayed register
R2	02H	Horizontal sync position register
R3	03H	Sync width register
R4	04H	Vertical total register
R5	05H	Vertical total adjust register
R6	06H	Vertical displayed register
R7	07H	Vertical sync position register
R8	08H	Interlace/skew register
R9	09H	Maximum scan line register
R10	0AH	Cursor start line register
R11	0BH	Cursor end line register
R12	0CH	High-order byte start address register
R13	0DH	Low-order byte start address register
R14	0EH	High-order byte cursor address register
R15	0FH	Low-order byte cursor address register
R16	10H	High-order byte light-pen register or mouse X counter register (neither is used in the computer)
R17	11H	Low-order byte light-pen register or mouse Y counter register (neither is used in this computer)

Horizontal Total Register (R0) — This 8 bit, write-only register is used to set the time between the starting points of character rows. This time, defined in character clock periods, includes the total time for all characters in a row, the delay before the horizontal sync pulse, the width of the sync pulse, and retrace time. Since the first character is 0 (zero), this value will be the number of character clock signals desired minus 1. A value of 112 would represent 113 character positions.

Horizontal Displayed Register (R1) — This 8-bit, write-only register is used to select the number of characters in each row that will be displayed on the screen. In normal operation, the register will contain a lower value than the horizontal total register (R0).

Horizontal Sync Position Register (R2) — This 8-bit, write-only register is used to determine the position of the sync pulse in each scan line. Any value greater than the horizontal displayed register (R1) and less than the horizontal total register (R0) can be used. When the value of R2 is increased, the display will shift to the left; decreasing this value shifts the display to the right.

Sync Width Register (R3) — This 8-bit, write-only register is used to set the width of the horizontal and vertical sync pulses. The width of the horizontal sync is controlled by the lower four bits of the pulse width register and the width of the vertical sync is controlled by the upper four bits of the pulse width register. Increasing the value lengthens the width.

NOTE: The sum of the values in register R1 and R3 must be equal to or less than the value in register R0.

Vertical Total Register (R4) — This 7-bit, write-only register is used to determine the refresh rate of the video monitor and vertical retrace timing. Its value is expressed in character rows; each is normally eight scan lines. The values for the various video modes are provided in Chapter 11.

Vertical Total Adjust Register (R5) — This 5-bit, write-only register is used to lengthen the vertical total time to account for the scan lines remaining when the value for register R4 is determined. This value is also expressed in scan lines and is provided in Chapter 11.

Vertical Displayed Register (R6) — This 7-bit, write-only register is used to determine the number of character rows that will be displayed. This value must be equal to or less than the vertical total register (R4).

31 kHz Video

Vertical Sync Position Register (R7) — This 7-bit, write-only register is used to control the position of the vertical sync. The value is expressed in character rows and must be equal to or less than the value in the vertical total register (R4). A lower value will move the display down; a higher value will move the display up.

Interlace/Skew Register (R8) — This 8-bit, write-only register is used to determine interlace operation (on or off) and the skew (offset or delay) of the characters and cursor. CGA operation lacks sufficient memory to operate the computer in an interlaced mode, and none of the PC-compatible modes use an interlaced mode. Therefore, always operate this video in a non-interlaced mode. Table 17-11 defines the functions of bits 0 and 1; Table 17-12 defines the functions of bits 4 and 5; and Table 17-13 defines the functions of bits 6 and 7; bits 2 and 3 are not used.

Table 17-11. 6845 Register R8, Bits 0 and 1

BIT 0	BIT 1	FUNCTION
0	0	Normal non-interlaced operation
0	1	Normal non-interlaced operation
1	0	Interlaced sync mode operation
1	1	Interlaced sync and video mode operation

Table 17-12. 6845 Register R8, Bits 4 and 5

BIT 4	BIT 5	FUNCTION
0	0	Normal display; no character skew.
0	0	Skewed display; characters are offset one character position.
1	0	Skewed display; characters are offset two character positions.
1	1	Illegal; do not use.

Table 17-13. 6845 Register R8, Bits 6 and 7

BIT 6	BIT 7	FUNCTION
0	0	Normal display; no cursor skew.
0	1	Skewed cursor; the cursor is offset one character position.
1	0	Skewed cursor; the cursor is offset two character positions.
1	1	Illegal; do not use.

Maximum Scan Line Register (R9) — This 5-bit, write-only register is used to control the number of scan lines in each character row. The first scan line is 0 so this value will be 1 less than the number of scan lines. Normally, eight scan lines are used for a character row; therefore, the value will be 7.

Cursor Start Line Register (R10) — This 7-bit, write-only register is used to define the first scan line of the cursor within the character field and whether the cursor is on or off.

The lower five bits define the first scan line. The scan lines are numbered from the top, starting with zero, to the bottom, ending with seven in the modes supported by this computer. Normally, the value in this register is 6.

Bit 5 must be clear (0) for the cursor to be on (blinking). Bit 6 may be set (1) or clear (0). The 6845 defines bits 5 and 6 to control the blinking (steady, 2 Hz, or 4 Hz).

Cursor End Line Register (R11) — This 5-bit, right-only register is used to define the last scan line of the cursor within the character field. Normally, this value is 7.

Start Address Registers (R12 and R13) — These two read/write registers contain the first displayable address in video memory and are used for scrolling the display. Register R12 contains the 6 bit, high-order (most significant byte) address, and register R13 contains the 8-bit, low-order (least significant byte) address.

31 kHz Video

Cursor Address Registers (R14 and R15) — These two read/write registers contain the address of the cursor in video memory. This allows the original cursor location to be retained even though the hardware may scroll or page the display. Register R14 contains the 6-bit, high-order (most significant byte) address and register R15 contains the 8-bit, low order (least significant byte) address.

Light Pen or Mouse Registers (R16 and R17) — These two read-only registers contain the video address captured from a light pen or the X and Y coordinates of a mouse. The mouse coordinates are relative to the last time the mouse coordinates were cleared. Neither of these applications are supported in this computer.

Video Mode Select Register

The video mode select port is actually a single 8-bit, read/write register. Its port address is 3D8H. Full CGA compatibility is maintained in this register. The bits of this register are somewhat interactive and are defined in Table 17-14 and Table 17-15.

Table 17-14. Video Mode Select Port Register

BIT	DESCRIPTION
0	Alphanumeric width: clear (0) = 40×25 , set (1) = 80×25 . Bit 1 must be clear (0). Bit 5 determines color/blink mode.
1	Alphanumeric/graphics mode: clear (0) = alphanumeric, set (1) = graphics. The width of the alphanumeric mode is determined by bit 0. The resolution of the graphics mode is determined by bits 4 and 6.
2	Color/monochrome composite mode: clear (0) = color, set (1) = monochrome composite.
3	Video enable/disable: clear (0) = disable video, set (1) = enable video. Video should be disabled (turned off) during mode changes to prevent an unprofessional display that results from synchronization changes that occur as modes are changed.
4	640×200 resolution color or monochrome/ 320×200 resolution. Refer to Table 17-15.

Table 17-14 (continued). Video Mode Select Port Register

BIT	DESCRIPTION
5	Alphanumeric color/blink mode: clear (0) = 16-color background enable, set (1) = blinking attribute enabled.
6	Color/Monochrome or 160 × 200/640 × 200 resolution. Refer to Table 17-15. This bit can be locked by software to prevent accidental modification by CGA software.
7	Active/standby: clear (0) = active, set (1) = standby (no output). This bit can be locked by software to prevent accidental modification by CGA software.

Table 17-15. Video Mode Select Resolution

BIT 4	BIT 6	DESCRIPTION
0	0	320 × 200 resolution
0	1	160 × 200 resolution
1	0	640 × 200 monochrome only resolution
1	1	640 × 200 16-color only resolution

Color Select Register

The color select port is a 6-bit, write-only register that establishes certain colors used by the various modes. Its port address is 3D9H. Because the colors selected can be modified by the 6845, the colors are represented by R, G, B, and I in Table 17-16 which describes the function of each bit of this register. Table 17-17 defines the two color palettes that can be used by the 320 × 200 graphics mode.

31 kHz Video

Table 17-16. Color Select Port Register

BIT	DESCRIPTION
0	B color, default is blue. Affects border and background color in 320 × 200 graphics mode, foreground color in 640 × 200 graphics mode, and border color in text modes. Bits 0, 1, 2, and 3 are combined to produce one of 16 programmable colors.
1	G color, default is green. Affects border and background color in 320 × 200 graphics mode, foreground color in 640 × 200 graphics mode, and border color in text modes. Bits 0, 1, 2, and 3 are combined to produce one of 16 programmable colors.
2	R color, default is red. Affects border and background color in 320 × 200 graphics mode, foreground color in 640 × 200 graphics mode, and border color in text modes. Bits 0, 1, 2, and 3 are combined to produce one of 16 programmable colors.
3	I signal, intensifies the color levels produced by the R, G, and B colors. Affects border and background colors in 320 × 200 graphics mode, foreground color in 640 × 200 graphics mode, and border color in text modes. Bits 0, 1, 2, and 3 are combined to produce one of 16 programmable colors.
4	When set (1), intensifies the foreground colors in the 320 × 200 graphics mode. Does not affect the text or 640 × 200 graphics modes.
5	320 × 200 graphics mode color palette selection: set (1) = palette 1, clear (0) = palette 2. Refer to Table 17-17 for palette definition. Does not affect the other modes.

Table 17-17. Palette Definitions

COLOR	PALETTE 1	PALETTE 2
0	See note 1	See note 1
1	G + B (cyan)	G (green)
2	R + B (magenta)	R (red)
3	R + G + B (white)	R + G (brown)

Table 17-17 (continued). Palette Definitions

COLOR	PALETTE 1	PALETTE 2
-------	-----------	-----------

NOTES: 1. If color 0 is called for, the color defined by bits 0-3 of the color select port register will be the color used.

2. The colors shown in parenthesis are the default colors produced by the 6845 for CGA operation. The 16 color palette registers can be modified to produce other colors, which will modify the colors in these palettes.

Status Register

The status port is actually a 5-bit, read-only register that reports the status of various features of the controller. Its address is at 3DAH. Table 17-18 describes what each bit means.

Table 17-18. Status Port Register Report

BIT	DESCRIPTION
0	Active display: set (1) = active display, clear (0) = vertical retrace (sync) period.
1	Not used or connected in this computer. In other hardware applications this bit can report that the light pen switch has been activated or that mouse switch 1 is closed.
2	Not used or connected in this computer. In other hardware applications, this bit can report the current status the light pen switch or that mouse switch 0 is closed.
3	Set (1) during the active vertical retrace period. Clear (0) during all other times. If the monochrome video mode is selected, which is not implemented in this computer, this bit will reflect the video dot stream, the same as bit 4.
4	The status port register bit 4 is designed to test the video mode select port, color select port, and all write-only registers.

31 kHz Video

Light Pen Register

The two light pen ports, addressed at 3DBH and 3DCH, are not implemented in this computer. The ports are actually write-only registers that are used to verify that the light pen trigger flip-flop and status read functions are operating correctly. Since light-pens are not supported by this computer, there is no light pen flip-flop and hence, these ports are meaningless.

MDA Interface

The following sections describe the programmable MDA characteristics emulated by the 31 KHz video card. They discuss specific registers required to produce the MDA format. A complete discussion of the 6845 CRTC registers is provided at the beginning of this section.

Control Register

The control register provides a number of functions. This register enables the CPU to communicate with the display RAM and attribute RAM, enables video data to be sent to the monitor, selects the font which is used, enables the blink attribute, and allows paging of the display. To control all of these functions, a single 8-bit byte is written into the control register. The control register is located at I/O port 3B8H. The function of each bit is shown in Table 17-19.

Table 17-19. MDA Control Register Functions

BIT NUMBER	BIT VALUE	FUNCTION
0	0	Prevents communication between the CPU and the display or attribute RAM. Data stored in RAM is displayed (if VIDEN is enabled) but cannot be changed.
	1	Enables communication between the CPU and the RAM. Data stored in RAM is displayed (if VIDEN is enabled). The RAM can be written to or read by the CPU.

Table 17-19 (continued). MDA Control Register Functions

BIT NUMBER	BIT VALUE	FUNCTION
1	X	Not used.
2	X	Not used.
3	0	Disables the display. The data stored in the display or attribute RAM is not changed but is not displayed. The RAM can be addressed by the CPU when the display is in this state (if the board is enabled).
	1	The display is enabled. Data stored in the display and attribute RAM is displayed.
4	0	This allows the first page of memory to be displayed.
	1	Pages the display to the second page of the display.
5	0	This allows the blink attribute bit in the attribute RAM to control the background intensity of the display.
	1	This allows the blink attribute bit to control the blinking of the character.
6/7	X	These bits control the font in use FONT 1 by the display. The font selection follows a standard binary count (00 = font 0, 01 = font 1, 10 = font 2, 11 = font 3).

NOTE: To initialize the card, write the bits 0001XX1 into I/O port 3B8.

Writing Data to the MDA Interface — The display is treated by the CPU as RAM. The text control data is written into the RAM on the Card. The CRTC and hardware on the card manage the display. The programmer has a number of options in the way that data can be handled.

31 kHz Video

NOTE: Bit 0 in the control register must be 1 to enable the write function. Before attempting to write to the memory, be sure this bit is properly set.

The programmer can alternately write data and control bytes to the display and attribute RAM, or fill the data into the display RAM and then fill the attribute RAM. The display RAM is enabled by a low on the CPU's A1 address line so that all attribute addresses are odd numbers. The addresses are shown in Table 17-20.

Table 17-20. Character/Attribute Addressing

DISPLAY LOCATION	CHARACTER ADDRESS	ATTRIBUTE ADDRESS
1	B0000	B0001
2	B0002	B0003
3	B0004	B0005
4	B0006	B0007
—	—	—
80 (last in line 1)	B00A0	B00A1
81 (1st in line 2)	B00A2	B00A3
—	—	—
1975 (1st in line 25)	B0F5E	B0F5F
2000 (last on page)	B0FA0	B0FA1

In most cases, the attributes are changed in blocks. This means that the same attribute code is entered repetitively until the end of the block is reached. When whole blocks of data are moved to the screen, it may be faster to enter all of the characters and then the attributes. In cases where single characters are entered from the keyboard, it may be easier to enter the character and then the attribute.

Attribute Codes — The 8 bits of data stored in the attribute RAM are used to control the display of each character in the display. Six functions are available for each character and selected by the data in the attribute RAM. These functions are shown in Table 17.21 along with the code that generates them. In order to blink the character, the blink enable bit in the control register must be set as described earlier in this chapter.

Table 17-21. Attribute Codes

FUNCTION	ATTRIBUTE DATA BIT								CONTROL REGISTER DATA BIT 3
	7	6	5	4	3	2	1	0	
Not displayed	0	0	0	0	0	0	0	0	x
Intensify background	1	x	x	x	x	x	x	x	0
Blink	1	x	x	x	x	x	x	x	1
Underline	x	x	x	x	x	0	0	1	x
Intensify foreground	x	x	x	x	1	x	x	x	x
Reverse video	x	1	1	1	x	0	0	0	x
x = value does not affect attribute									

Cursor Control — The cursor is generated by the CRTC. The current location of the cursor can be determined by reading the contents of CRTC register R14 and R15. The cursor can be moved by writing a new location into the registers.

The value returned from this register will not be the same as the address the CPU uses to address the identical display location. The CPU must address the display and attribute memories at different locations. The CRTC addresses them at the same time. Bit 0 of the CPU address is used to select which of the two memories is being talked to. Since the CRTC addresses the two RAM's at the same time, there is no corresponding bit 0. Character address bit 0 from the CRTC is the same as address bit 1 from the CPU. The relationship of the addresses is such that the address from the CRTC registers must be shifted left one position to make them agree.

31 kHz Video

To reposition the cursor, do the following:

1. Determine the location you want the cursor to move to.
2. Write 0EH into I/O address 3B4H to address the high-order cursor address (CRTC index register R0).
3. Write the high-order value of the cursor address to I/O address 3B5H. Remember that the CRTC address is shifted to the right one position from the CPU address.
4. Write 0FH into I/O address 3B4H to address the low-order cursor address register (CRTC register R15).
5. Write the low-order address of the cursor to I/O address 3B5H.

Blanking/Clearing the Display — Blanking the display means that the characters stored in the character memory are not displayed on the CRT. The memory is not affected and can be read or changed while the screen is blanked. The display is cleared when all data stored in the RAM is erased.

The display is blanked through the video control register. To blank the display, write XXXX 0XX1 (for example, 01H) to I/O address 3B8H. No characters will be displayed on the CRT but the RAM may be changed at this time. Clearing the display requires writing 0s to all RAM locations.

Scrolling the Display — The programmer can use the flexibility of the 6845 CRTC to scroll the display. The starting point of the display can be moved to a new location in the RAM without changing the RAM. This is done by writing 0CH into I/O address 3B4H to prepare the high-order register in the 6845 to accept data. Then write the high-order location to I/O address 3B5. The next step is to write 0DH to I/O address 3B4H and then the low-order location into I/O address 3B5H. The top-left corner of the display will now be the address specified. The programmer must exercise caution when addressing starting locations beyond the first page so that no addresses beyond the capability of the RAM are used at the end of the page.

Paging the Display — The standard RAM contains enough memory to display four pages of standard text. The programmer can page in one of two ways: using the CRTC address registers as described for scrolling (enter an address one page higher than the current address instead of one line higher), or through the video control register.

To use the video control register to page the text, write a 1 into bit 4 of the byte at I/O address 3B8H. Be careful not to change the other bits of this byte when using this procedure.

Character Font — Table 17-22 illustrates the supported character font.

Table 17-22. Character Font

DECIMAL VALUE	HEX-DECIMAL VALUE	0	16	32	48	64	80	96	112
0	1	2	3	4	5	6	7		
0	0	BLANK (SPACE)	BLANK (SPACE)	0	@	P		p	
1	1	☺	☹	!	1	A	Q	a	q
2	2	☻	☼	"	2	B	R	b	r
3	3	♥	♠	#	3	C	S	c	s
4	4	♦	♣	\$	4	D	T	d	t
5	5	♣	♠	%	5	E	U	e	u
6	6	♠	♣	&	6	F	V	f	v
7	7	•	•	'	7	G	W	g	w
8	8	•	•	(8	H	X	h	x
9	9	○	○)	9	I	Y	i	y
10	A	○	○	*	:	J	Z	j	x
11	B	♂	♀	+	;	K	[k	{
12	C	♀	♂	,	<	L	\	l	
13	D	♂	♀	—	=	M]	m	}
14	E	♂	♀	.	>	N	^	n	~
15	F	☼	☹	/	?	O	_	o	Δ

DECIMAL VALUE	HEX-DECIMAL VALUE	128	144	160	176	192	208	224	240
0	1	2	3	4	5	6	7	8	9
0	0	℄	℄	á	á	á	á	∞	≡
1	1	ü	Æ	í	í	í	í	β	±
2	2	é	FE	ó	ó	ó	ó	γ	≥
3	3	â	ô	ú	ú	ú	ú	π	≤
4	4	ä	ö	ñ	ñ	ñ	ñ	Σ	∫
5	5	à	ò	ñ	ñ	ñ	ñ	σ	∫
6	6	â	û	ä	ä	ä	ä	μ	÷
7	7	ç	ù	o	o	o	o	τ	≈
8	8	ê	ÿ	ç	ç	ç	ç	Φ	°
9	9	ë	Ö	Γ	Γ	Γ	Γ	Θ	•
10	A	è	Û	Γ	Γ	Γ	Γ	Ω	•
11	B	ï	€	½	½	½	½	δ	√
12	C	î	℄	¼	¼	¼	¼	∞	η
13	D	ï	℄	ï	ï	ï	ï	Ø	²
14	E	Ä	Pls	«	«	«	«	Ε	■
15	F	Ä	f	»	»	»	»	∩	Blank

31 kHz Video

Status Register

The status register is a 4-bit, read-only register located at I/O port 3BAH. Bit 0 is the horizontal drive signal, bits 1 and 2 are the font number in use, and bit 3 is the video data. The programmer can look at bits 1 and 2 for font information and can use bits 0 and 3 for diagnostics, but data bits 0 and 3 are read "on the fly". These data bits may not meet the timing requirements to be read properly.

Quick Reference Charts — The following tables give the addresses and default values for the registers accessible to the programmer. Table 17-23 shows all the addresses which can be written to or read from and their use. Table 17-24 shows the CRTC registers and their default values.

Table 17-23. MDA Register Addresses

I/O ADDRESS	FUNCTION	DESCRIPTION
3B4	CRTC Register	Index Used to select a CRTC register to write to. This register must be written to before data can be entered into the CRTC registers.
3B5	CRTC Register	Data Data written into this register is loaded into the CRTC register selected by writing into address 3B4.
3B8	CRT Control Register	This register controls the functions of the card including communication, video display, the page of the display, blink enable, and font selection. The function of these bits is shown in Table 17-19.
3BA	CRT Status Port	This port allows the programmer to look at the font selected, the video data, and the horizontal sync.

Table 17-24. CRTC Registers and Default Values

REGISTER #	REGISTER FILE	PROGRAM UNIT	80 × 25 MONOCHROME
R0	Horizontal Total	Characters	61H
R1	Horizontal Displayed	Characters	50H
R2	HSYNC Position	Characters	52H
R3	HSYNC Width	Characters	FH
R4	Vertical Total	Char Rows	19H
R5	VTOTAL Adjust	Scan Line	6H
R6	Vertical Displayed	Char Rows	19H
R7	VSYNC Position	Char Rows	19H
R8	Interlace Mode	—	02
R9	Max Scan Line Address	Scan Line	DH
R10	Cursor Start	Scan Line	BH
R11	Cursor End	Scan Line	CH
R12	Start Address (H)	—	00H
R13	Start Address (L)	—	00H
R14	Cursor (H)	—	00H
R15	Cursor (L)	—	00H
R16	Reserved	—	—
R17	Reserved	—	—

CGA Interface

The following sections describe the programmable CGA characteristics emulated by the 31 KHz video card. They discuss specific registers required to produce the CGA format. A complete discussion of the 6845 CRTC registers is provided at the beginning of this section.

CRT Controller Registers

The user can program the 19 internal registers of the 6845 CRT controller to define display and control parameters of a raster-scanned monitor. One of these registers, the address register, is used only as a pointer to the addresses of the other 18 registers. The address register is write-only. The CPU loads this register using an I/O OUT instruction to I/O address 3D4. This register is loaded with the five least-significant bits of the I/O argument.

31 kHz Video

The other 18 registers are selected by the pointer information in the address register, then the CPU directs the information at address 3D5 to be loaded into the selected register. Transfers of address and parameter information to and from the registers is via data lines D0 through D7.

Table 17-25 defines the individual registers in terms of values contained in them and the results obtained with these values. All values are expressed in hexadecimal.

Table 17-25. CRT Controller Register Functions

REG. ADDR.	REG. NO.	REG. TYPE	UNITS	REG. STAT.	VALUES		
					40 x 25 TEXT	80 x 25 TEXT	GRAPHICS MODES
0	R0	Horiz. Total	Char.	Write	38	71	38
1	R1	Horiz. Dispd.	Char.	Write	28	50	28
2	R2	Horiz. Sync. Position	Char.	Write	2D	5A	2D
3	R3	Horiz. Sync.	Char.	Write	0A	0A	0A
4	R4	Vert. Total	Row	Write	1F	1F	7F
5	R5	Vert. Total Adjust	Scan Line	Write	06	06	06
6	R6	Vert. Dispd.	Row	Write	19	19	64
7	R7	Vert. Sync. Position	Row	Write	1C	1C	70
8	R8	Interlace Mode	—	Write	02	02	02
9	R9	Max. Scan Line Addr.	Scan Line	Write	07	07	01
A	R10	Cursor Start	Scan Line	Write	06	06	06

Table 17-25 (continued). CRT Controller Register Functions

REG. ADDR.	REG. NO.	REG. TYPE	UNITS	REG. STAT.	VALUES		
					40 × 25 TEXT	80 × 25 TEXT	GRAPHICS MODES
B	R11	Cursor End	Scan Line	Write	07	07	07
C	R12	Start Addr.	High	Write	00	00	00
D	R13	Start Addr.	Low	Write	00	00	00
E	R14	Cursor Addr.	High	Read/ Write	XX	XX	XX
F	R15	Cursor Addr.	Low	Read/ Write	XX	XX	XX
10	R16	Light Pen	High	Read	XX	XX	XX
11	R17	Light Pen	Low	Read	XX	XX	XX

The first group of registers, R0 through R3, establishes the horizontal format and timing parameters. Registers R4 through R9 determine vertical format and timing parameters. The remaining registers, R10 through R17, deal with cursor characteristics, screen memory addressing, and the light pen interface. Typically, registers R0 through R11 are loaded when the system is turned on and should not need to be accessed thereafter. The other six registers, R12 through R17, need to be accessed on an ongoing basis during display operations. R12 and R13 establish the 14-bit starting (top-of-page) address of screen memory. The contents of these registers may be manipulated to perform scrolling of the screen contents. R14 and R15 establish the 14-bit cursor address which establishes the cursor symbol position on the screen. R16 and R17 are used only if a light pen is interfaced.

31 kHz Video

Horizontal Timing and Format Registers — The contents of register R0 determines the total time allotted for one scan line in terms of character clock (CLK) cycles (total of displayed and undisplayed characters minus 1) per horizontal line, thus determining the horizontal sync (HSYNC) frequency.

R1 is the character row register. It is loaded with the total number of characters minus 1 in character clock (CLK) units.

The horizontal sync register, R2, establishes the point at which the HSYNC signal makes its negative-to-positive transition, specified in terms of CLKs. The reference point for the beginning of the HSYNC pulse is the left-most character position displayed on the scan line.

R3, the HSYNC width register, uses only the four least-significant bits of data to establish the duration of the HSYNC pulse in the range of 1 to 16 character clocks. This allows you to adjust the HSYNC pulse duration to the requirements of the applicable CRT monitor.

Figure 17-3 shows the inter-relationship between the four horizontal timing and format registers (R0 through R3) in terms of CLK.

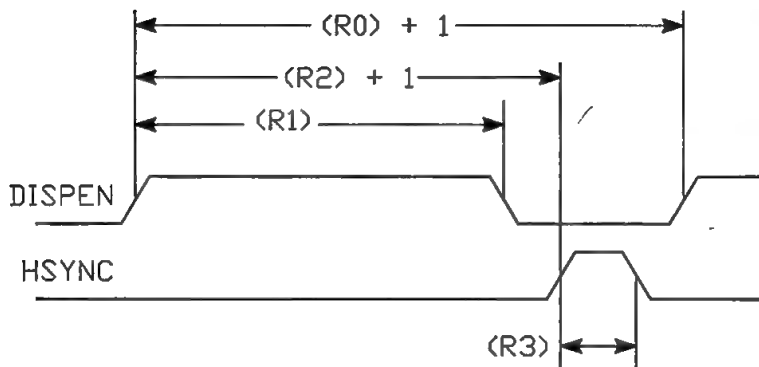


Figure 17-3. Horizontal Timing and Format

Vertical Timing and Format Registers — Registers R4 through R9 normally are loaded at system startup and normally are not changed thereafter. The point of reference for these registers is the top-most character position displayed on the monitor screen.

The vertical total register, R4, and the vertical sync adjust register determine the total number of scan line times in a frame, including vertical retrace, establishing the overall frame rate or VSYNC frequency.

R4 is a 7-bit register loaded with the total number of character rows. Since a character row can consist of up to 32 scan lines, it may be difficult for you to establish a refresh frequency close to the line frequency. Register R5, a 5-bit VSYNC adjust register, may then be used to fine-tune the VSYNC frequency. R5 is loaded with a value representing scan line times.

The character-rows-displayed register, R6, is a 7-bit register which allows you to select the number of rows of characters to be displayed, up to 128. What you specify for this register does not determine the positions of the VSYNC pulse, but the point at which display enable (DISPEN) will be reset for vertical retrace.

R7, the vertical sync (VSYNC) position register, determines the point at which the VSYNC signal makes its negative-to-positive transition to initiate vertical retrace. VSYNC position is determined by the character row times, measured from the first character row on the monitor screen. The VSYNC pulse always has a duration of 16 scan lines. Since the scan line frequency will vary from one application to another, and since the VSYNC pulse duration cannot be changed, external circuitry may be needed to achieve a pulse that is compatible with the monitor used.

The interlace mode register, R8, determines whether interlaced or non-interlaced scan is used.

The value in register R9 determines the number of scan lines per character row. This 5-bit register may be programmed for a character row of up to 32 scan lines. The value used will dictate the maximum count output by the raster address signals (RA0 through RA4) to the character generator logic. Load R9 with the desired scan line count minus 1.

31 kHz Video

The cursor formatting registers are comprised of the cursor start and cursor stop registers, R10 and R11, respectively. The five least-significant bits of each register determine the scan lines within a character row where the cursor symbol is to be activated. The scan line specified in R10 is the first scan in which the CURSOR signal is to be set and it will remain set until the scan line specified in R11 has been completed. Accordingly, if the cursor symbol is to occupy a single scan line, the same value must be loaded into both registers. If different values are loaded, a block-type cursor will be formed. In interlaced sync and video mode, both registers must be loaded with either odd or even values. Bits 5 and 6 determine whether or not the cursor is to blink, and if so at either 1/16th or 1/32th of the field rate. See the data sheets in Chapter 8 for clarification of cursor format programming.

Primary Operating Registers — The remaining six registers, R12 through R17, are considered the primary operating registers since, in the course of display programming, the contents will undoubtedly change on an ongoing basis rather than being loaded one-shot at system startup. The six registers are arranged as three 14-bit register pairs (bits 6 and 7 of the most-significant byte are not used).

R12 and R13 comprise the top-of-page register which specifies the screen memory address containing the first character from the top-left corner to be displayed. At the end of each vertical retrace, the first screen memory address generated will be the one contained in these registers. Since the 6845 addresses memory linearly, rather than on a row/column basis, scrolling can be performed on either a character-by-character or row-by-row basis.

The cursor position register, R14 and R15, generate a cursor signal when the 6845 generates a screen memory address on MA0 through MA13 that matches the contents of this register pair, and when the scan line counter (RA0 through RA4) outputs fall within the limits established by registers R10 and R11. It may be necessary to delay the cursor signal with external logic circuitry to achieve display at the desired position. Cursor movement on the screen is accomplished by loading new values into R14 and R15. These registers are the 6845's only read/write registers so they also can be used to keep track of the cursor position, rather than copying this information from another memory location.

R16 and R17 are the light pen register pair, which will be loaded with the screen memory address corresponding to the screen position at which the LPEN signal was detected. Since there are several critical timing parameters involved with this signal, be sure to carefully study the documentation supplied with this option.

Raster Scan Registers — The raster scan outputs, RA0 through RA4, provide the interface between the 6845 and the character generator logic. These outputs may represent scan line counts of from 0 to 31, or up to 32 scan lines per character row. Register R9, the scan lines/row register, determines the maximum count from the scan output registers before reset occurs. Scan line counts are incremented at the HSYNC rate, but also are dependent on the interlace format selected.

I/O Registers — Table 17-26 defines the CGA I/O devices emulated by the 31 KHz video card.

Table 17-26. CGA I/O Port Assignments

PORT ADDRESS	REGISTER
3D0	6845
3D1	6845
3D8	Mode Control
3D9	Color Select
3DA	Status/Mode I/O
3DB	Light Pen Latch CLEAR
3DC	Light Pen Latch PRESET

31 kHz Video

To address a given port, the address lines are programmed as defined in Table 17-27.

Table 17-27. CGA I/O Port Selection

PORT	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
3D0	1	1	1	1	0	1	1	X	X	0
3D1	1	1	1	1	0	1	1	X	X	1
3D8	1	1	1	1	0	1	1	0	0	0
3D9	1	1	1	1	0	1	1	0	0	1
3DA	1	1	1	1	0	1	1	0	1	0
3DB	1	1	1	1	0	1	1	0	1	1
3DC	1	1	1	1	0	1	1	1	0	0

X = Don't care

The 6845 CRT controller occupies two I/O port addresses. The RS (register select) line is connected to the least-significant bit, A0. When A0 is reset, the address register is accessed to load it with the register address of the desired parameter, which would then be accessed by setting A0 (RS). A typical register access operation then would consist of two consecutive device write cycles, or a write cycle followed by a read cycle, and these cycles would be directed to consecutive memory or I/O locations.

Color Select Register — The color select register, located at address 3D9, is a 6-bit output-only device (it cannot be read) which can be written to using the I/O OUT command. Only the six least-significant bits of the instruction byte are used, as shown in Table 17-28.

Table 17-28. Color Select Register Logic

BIT	DESCRIPTION
0	B (blue) border color select (text mode) or 640 × 200 graphics dot color.
1	G (green) border color select (text mode) or 640 × 200 graphics dot color.
2	R (red) border color select (text mode) or 640 × 200 graphics dot color.
3	I (intensity) intensifies border color (text mode) or 640 × 200 graphics dot color.
4	Selects alternate background color palette for text mode color.
5	320 × 200, graphics color palette selection.
6 and 7	Not used.

In text mode, bits 0, 1, 2, and 3 determine the screen border color. In 320 × 200 graphics submode, these bits select the screen background color (C0 and C1).

Bit 4, when set, selects an alternate, intensified palette of background colors in text mode if bit 5 in the mode select register, I/O port 3D8, is reset.

31 kHz Video

Bit 5 is used only in the 320 × 200 graphics submode to select the active palette of screen display colors. If bit 5 is a logical 1, color is determined by the logic of Table 17-29. If bit 5 is a logical 0, color is determined by the logic of Table 17-30.

Table 17-29. Palette Number 1 Selection

C1	C0	BACKGROUND COLOR
0	0	Defined by bits 0-3 of I/O Port 3D9
0	1	Cyan
1	0	Magenta
1	1	White

Table 17-30. Palette Number 2 Selection

C1	C0	BACKGROUND COLOR
0	0	Defined by bits 0-3 of I/O port 3D9
0	1	Green
1	0	Red
1	1	Yellow

Mode Select Registers — The mode select registers are also write-only registers. One is a 6-bit register at port address 3D8; the other is an 8-bit register at 3DA. Both can be written to using I/O OUT commands. The output register at 3D8 functions according to the logic of Table 17-31.

Table 17-31. Mode-Select Port 3D8 Logic

BIT	DESCRIPTION
0	A logical 1 selects 80 × 25 text submode. A logical 0 selects 40 × 25 submode.
1	A logical 1 selects 320 × 200 graphics submode. A logical 0 selects text mode.
2	A logical 1 selects monochrome submode. A logical 0 selects color submode.
3	Disables the video signal during mode changes. The video signal should be re-enabled following a mode change. A logical 1 enables the video.
4	A logical 1 selects 640 × 200 monochrome graphics submode. Use I/O port 3D9 to select one of eight colors when using a direct drive monitor in this submode.
5	A logical 1 selects blinking feature in alphanumeric mode (normal setting). A logical 0 intensifies whichever of the 16 background colors is selected to yield a lighter shade of that color.
6 and 7	Not used.

31 kHz Video

The register at 3DA functions according to the logic of Table 17-32.

Table 17-32. Mode-Select Port 3DA Logic

BIT	DESCRIPTION	
0	A logical 1 overrides the VIDEO ENABLE signal from 3D8; will eliminate monitor display flicker when in text mode.	
1	Character font selection. A logical 1 selects the single-dot width font. A logical 0 selects the double-dot width font. The purpose for allowing this either/or situation is to enable your selection of either font at power on.	
2	A logical 1 will underline any character which has a logic one in bit 6 (RED background) of the attribute byte.	
3	Reserved for video diagnostics.	
4	Reserved.	
5	A logical 1 enables fast hardware scrolling in the 80 × 25 sub-mode only.	
6 and 7	Used to select one of four pages of display text from the video RAM according to the following logic:	
<u>BIT 7</u>	<u>BIT 6</u>	<u>SCREEN PAGE SELECT</u>
0	0	First page
0	1	Second page
1	0	Third page
1	1	Fourth page

Status Register — The status register, an 8-bit read-only buffer, resides at input port address 3DA. Table 17-33 summarizes the logical operation of this register.

Table 17-33. Status Select Logic

BIT	DESCRIPTION
0	A logical 1 indicates that a horizontal or vertical retrace is in progress. This bit can be used to update the video memory during screen refresh intervals.
1	A logical 1 in this bit position indicates that a positive-going light pen signal has set the light pen trigger. This trigger is reset at power up and also may be cleared by issuing an I/O OUT command to port address 3DB. The reset is address-activated; no specific data need be output.
2	A logical 0 indicates that the light pen switch is on. The switch signal is not latched or debounced.
3	A logical 1 indicates that a vertical retrace is in progress.
4	Least-significant bit (LSB) of the character font number.
5	Most-significant bit (MSB) of the character font number.
6 and 7	Not used.

Programming Sequence

In general, observe the following sequence when selecting operating parameters for the video interface:

1. Determine desired mode of operation: text mode or all-points-addressable graphics.
2. Reset video enable bit. Also reset video override bit 0 in port address 3DA if not already reset.
3. Program the CRT controller to desired mode.
4. Determine desired submode(s), then program mode and color select registers.
5. Set video enable bit. Also set the video override bit, if desired.

HGC Interface

The following sections describe the programmable HGC characteristics emulated by the 31 kHz video card. They discuss specific registers required to produce the HGC format. A complete discussion of the 6845 CRTC registers is provided at the beginning of this section.

Index and Data Registers

The index and data registers are I/O ports that can be programmed to interface the HGC format to the proper type of video monitor. The index (or pointer-address port) register is addressed at port address 3B4H and the data port is addressed at 3B5H.

The index register acts as a gateway to 18 data registers. The values placed in the data registers establish the frequency for the monochrome display and the width and height of the display. Various other parameters can be established through the data registers. A complete analysis of these parameters is given in the "6845 CRT Controller Emulation" discussion at the beginning of this section.

Display Mode Control Register

This port is an output port that sets a specific mode of operation. The display mode control register is addressed at port address 3B8H.

CAUTION

Be very careful when programming this port. If an incorrect video mode is accidentally programmed, it could permanently damage the video monitor.

Table 17-34 details each bit's function of the display mode control register.

Table 17-34. Display Mode Control Register

BIT	DESCRIPTION
0	Reserved.
1	The state of this bit selects between the text or graphics mode of operation. If this bit is set low (zero), the text mode is active. If this bit is set high, the graphics mode is active. The powerup default provides the text mode of operation. Whenever a mode change occurs, the 6845 CRTC must be reprogrammed accordingly.
2	Reserved.
3	The state of this bit either blanks the monitor's screen or enables it. When set low, the screen is blanked and when set high, the screen is enabled. The powerup default provides a blanked screen. This function is useful when modes are changed. Blanking the screen allows the transition between modes to occur without possible screen bounce.
4	Reserved.
5	The state of this bit enables or disables the text blinking attribute that is assigned to any one character. When this bit is set low, the blinking attribute is disabled and when set high, the characters that have the attribute assigned to them will blink. Note that the blinking attribute does not affect the screen cursor. The powerup default disables blinking.
6	Reserved.
7	The state of this bit determines the starting memory address for the display buffer. When set low, the display buffer is addressed at B0000H (page 0) and when set high, the display buffer is addressed at B8000H (page 1).

CAUTION

Be particularly careful when you change from text to graphics or from graphics to text modes. The following simultaneous program sequence is recommended:

- Specify the desired mode (text or graphics) by setting bit 1 of the control mode register to the appropriate state. Do not use high-level programming languages to change values in the display mode control register. Random vertical and horizontal sync frequencies may result that could damage the monitor.
- Reprogram the 6845 CRT controller data registers with the appropriate values. Refer to the "6845 CRT Controller Emulation" discussion at the beginning of this section.

Display Status Register

This port is an input port that receives continual display status. The register in this port is addressed at port address 3BAH. Table 17-35 details each bit's function of the display status port.

Table 17-35. Display Status Register

BIT	DESCRIPTION
0	When the state of this bit is low (zero), it indicates that there are normal characters being output to the video monitor. When the state of this bit is high, it indicates that horizontal sync is occurring and that the normal character output is blanked (temporarily disabled). The persistence of the display maintains the brightness of the characters during sync periods.

Table 17-35 (continued). Display Status Register

BIT	DESCRIPTION
1	Reserved.
2	Reserved.
3	The state of this bit gives the status of the display with respect to the video signal. When this bit is low, it indicates that the video signal is inactive and no video "dots" are being sent to the display. When this bit is high, it indicates that the video signal is actively sending video dots to the display.
4	Reserved.
5	Reserved.
6	Reserved.
7	When the state of this bit is low, it indicates that vertical retrace is occurring and that no characters are being output to the video monitor (vertical blanking). When the state of this bit is high, it indicates that characters are being output to the video display and that an active trace period is occurring. The persistence of the display maintains the brightness of the characters during sync periods.

Configuration Register

This port permits configuration of the video mode and selects the desired video display buffer (page). The register in this port is addressed at port address 3BFH. The status of the bits in this register must be set in accordance with the appropriate bits of the display mode control register (3B8H) to select the desired display buffer. Table 17-36 details each bit's function of the 2-bit configuration port.

Table 17-36. Configuration Register

BIT	DESCRIPTION
0	When the state of this bit is low (zero), the graphics mode is disabled. Bit 1 of the display mode control register is forced low. When the state of this bit is high, the graphics mode is enabled. The powerup default of this bit disables the graphics mode (set low).
1	The state of this bit determines which display buffer is active. When the state of this bit is low, the active display buffer resides at address B0000H (page 0). When the state of this bit is high, the active display buffer resides at address B8000H (page 1).

The emulated Hercules Graphics Card format produces text mode video by storing character and attribute codes in a 4K display buffer. The codes that define each character are stored in the display buffer contiguously. 160 bytes (one for the character code and one for the attribute code) are required to create enough characters to fill one entire line on the video monitor.

The character generator can produce 256 unique characters. Each 8-bit character code contains the information required to define a specific character. The character set is identical to the character set produced by a typical PC character generator. However, the character font is a double-dot font displayed in a 9-pixel × 14-pixel character cell.

The attribute code is decoded to highlight the selected characters with a video attribute. The attributes include character underlining, blinking, boldface, reverse video, and blank characters. Table 17-37 details which attribute codes produce each video attribute.

Table 17-37. Video Attributes

7	6	5	BIT				0	ATTRIBUTE
			4	3	2	1		
X	0	0	0	1	0	0	0	Blank character
X	0	0	0	1	0	0	1	Underline
X	0	0	0	1	1	1	1	Normal
X	1	1	1	1	0	0	0	Reverse video

NOTES

1. If the status of bit 3 (1) is high, the character will appear in boldface.
2. If the status of bit 7 (X) is high, and the status of bit 5 in the display mode control register is low, the character background will appear in boldface.
3. If the status of bit 7 (X) is high, and the status of bit 5 in the display mode control register is high, the character background will blink.

Graphics are displayed when the display mode control register and the configuration register are programmed accordingly. The graphics mode is bit-mapped so that each bit of video data controls one picture element (pixel). The display buffer for the graphics mode consists of two separate memory regions (pages). The first memory region (page 0) is addressed at B0000H – B7FFFH. Page 1 is addressed at B8000H – BFFFFH.

Only one video page can be displayed at a time. Dynamic manipulation of the video screen can be achieved through altering values in the display buffer. Changes made to the buffer are immediately displayed. The inactive buffer can also be manipulated. The changes made to that buffer will not be seen until the buffer is made active. The display mode control register must be programmed to select the active display buffer (video page).

Enhanced Graphics

The 31 kHz video card emulates all of the functions provided by the EGA video interface. This card supports enhanced graphics on 31 kHz horizontal frequency monochrome and color monitors. The emulated EGA modes include 720-pixel by 350-pixel monochrome high-resolution graphics, 16-color graphics with either 320-pixel by 200-pixel or 640-pixel by 200-pixel resolution, and alphanumeric and bit-mapped, 16 color (out of a palette of 64 possible colors) 640-pixel by 350-pixel resolution for enhanced color display.

In the alphanumeric modes, characters may be selected from either of the predefined 256-member character generators. One generator defines 7-pixel by 9-pixel characters in a 9-pixel by 14-pixel character cell; the other defines 7-pixel by 7-pixel characters in an 8-pixel by 8-pixel character cell. Two additional 256-member character sets may be defined by the user and accessed at the same time, allowing the user to select any of over 1000 characters to be displayed at any time.

The following paragraphs provide a description of the major devices on the 31 kHz card that implement EGA emulation.

Card and EGA addressing and control signals are received from the system and sent through the ROM decoder circuits to the other elements of the card. Bidirectional data flow is handled by the CRT and graphics controllers. The sequencer controls the flow of data between the CPU and video circuits and the attributes controller combines the characters generated with the various video attributes to produce the output sent to the video display. The multiplexer allows graphics and character data and control functions to be multiplexed together, simplifying the wiring of the card.

CRT Controller — The CRT controller generates all horizontal and vertical sync timing signals, address signals for accessing and displaying the contents of video memory, cursor and underline timing signals, and the refresh address signals for the dynamic RAM.

Sequencer — The sequencer generates the timing signals for use by the video memory as well as the character clock for controlling all video memory fetches. The sequencer also allows the host computer to access video memory by inserting dedicated processor memory access cycles between the video memory display cycles (during horizontal and vertical retrace).

Graphics Controller — The graphics controller directs the data from the host computer's memory to the attribute controller and the processor. In graphics modes, data is sent in serial form to the attribute controller. In alphanumeric modes, the memory data is sent in parallel form directly to the video card's memory, bypassing the graphics controller. The graphics controller also formats the data for normal color modes when requested.

Attributes Controller — The attributes controller provides a color palette of 16 colors, each of which may be separately specified from 64 possible colors. The controller receives data from the video card's memory and formats it for display on the CRT screen. This device also controls cursor and character blinking and underlining.

Memory Planes — The display buffer is the video memory on the EGA card, and it contains 256K of dynamic RAM configured as for 64K memory planes of memory. The base address of this memory is set at A0000H to remain compatible with other video cards and applications software.

ROM — The firmware on the EGA card is designed for use with the computer system's firmware. It is unique to Zenith Data Systems computers and may not be fully compatible with other manufacturers' computers. This ROM contains the character generators and control code. It is mapped into the system memory at C0000H and occupies 16K.

Support Logic — The support logic provides multiplexing circuits and latches for the CRT controller, processor, and character generator. Two clock sources (14 MHz and 16 MHz) determine the timing and dot clock rate. The clock is multiplexed under processor control.

31 kHz Video

The 31 kHz EGA card's circuits are built around a chip set of four very large scale integrated circuits (VLSI). The set consists of an 82C431 Graphics Controller, and 82C432 Sequencer, and 82C433 Attributes Controller, and an 82C434 CRT Controller. This set supports color and monochrome (TTL-compatible) monitors in alphanumeric and bit-mapped (all-points-addressable) modes, including those supported by the IBM color graphics adapter and monochrome display adapter. The various video modes are described later in this manual.

The following paragraphs, tables, and illustrations describe the function of each IC, its registers, and operation.

82C431 Graphics Controller

The graphics controller directs data from the video memory to the attributes controller and system's CPU. It operates in two modes: alphanumeric mode and graphics mode. In the alphanumeric mode, data is sent in parallel form from video memory directly through the graphics controller to the attributes controller. In the graphics mode, data from video memory is converted to serial form before it is sent to the attributes controller.

The graphics controller also formats the data for use in the various video modes. It provides color comparators that are used in color painting. Built-in logic functions manipulate the data before it is written to video memory so that it can be written in 32-bit words, expediting video speed.

The controller is divided into two sections called Graphics A and Graphics B, each of which has an internal register assigned to it. Graphics A writes to memory planes 0 and 1 and Graphics B writes to memory planes 2 and 3.

Table 17-38 provides a summary of the registers in the Graphics Controller

Table 17-38. Graphics Controller Register Summary

PORT ADDRESS	REGISTER NUMBER	REGISTER NAME
3CC	—	Graphics A position
3CA	—	Graphics B position
3CE	—	Address register (00 – 08)
3CF	R0	00 — set-reset
3CF	R1	01 — enable set-reset
3CF	R2	02 — color compare
3CF	R3	03 — data rotate
3CF	R4	04 — read map select
3CF	R5	05 — mode
3CF	R6	06 — miscellaneous
3CF	R7	07 — color don't care
3CF	R8	08 — bit mask

Graphics A Position Register — The Graphics A position register may be programmed with an 0 or a 1. For normal operation, the Graphics A position register must be programmed with a 0 and the Graphics B position register must be programmed with a 1. This will direct data bus signals D0 to memory plane 0 and D1 to memory plane 1.

Although you can program this register to route D0 to memory plane 2 and D1 to memory plane 3, the EGA card is not designed to operate in this manner.

Graphics B Position Register — The Graphics B position register may be programmed with a 0 or a 1. For normal operation, the Graphics B position register must be programmed with a 1 and the Graphics A position register must be programmed with a 0. This will direct data bus signals D2 to memory plane 2 and D3 to memory plane 3.

Although you can program this register to route D2 to memory plane 0 and D3 to memory plane 1, the EGA card is not designed to operate in this manner.

31 kHz Video

Address Register — The address register is used to point to the other registers in the 82C431 and route input and output to the indicated register through port 3CFH. The four least significant bits determine the register selected. Table 17-39 defines the address of each register.

Table 17-39. Graphics Controller Addressing

BIT 3	BIT 2	BIT 1	BIT 0	VALUE	REGISTER
0	0	0	0	00	R0 — set-reset register
0	0	0	1	01	R1 — enable set-reset register
0	0	1	0	02	R2 — color compare register
0	0	1	1	03	R3 — data rotate register
0	1	0	0	04	R4 — read map select register
0	1	0	1	05	R5 — mode register
0	1	1	0	06	R6 — miscellaneous register
0	1	1	1	07	R7 — color don't care register
1	0	0	0	08	R8 — bit mask register

The partial program in Listing 17-1 illustrates a method that you may use to place a value (previously loaded into the CPU register BL) into the data rotate register of the 82C431 graphics controller. Loading data into other registers follows a similar pattern.

Listing 17-1. 82C431 Programming Example

```

                                ;select the register
MOV DX, 3CEH                   ;port address of the address register
MOV AL, 03                     ;select the register for the data
OUT DX, AL                     ;send AL to port defined by DX
                                ;sent the data

MOV DX, 3CFH                   ;port address for internal register
MOV AL, BL                     ;place value previously loaded in BL into AL
OUT DX, AL                     ;send value to register

```

Set-Reset Register (R0) — The value placed in bits 0 – 3 of this register will be written to the corresponding memory plane. To write the value to video memory, the mode register (R5) must have the write mode selected and the corresponding enable bits in the enable set-reset register (R1) selected. Table 17-40 describes the bit to memory plane relationship.

Table 17-40. Bit to Memory Plane Relationship

BIT	MEMORY PLANE
0	If set to 1, set memory plane 0; if set to 0, reset memory plane 0.
1	If set to 1, set memory plane 1; if set to 0, reset memory plane 1.
2	If set to 1, set memory plane 2; if set to 0, reset memory plane 2.
3	If set to 1, set memory plane 3; if set to 0, reset memory plane 3.

Enable Set-Reset Register (R1) — This register works with R0 (the set-reset register) and R5 (the mode register). In order to enable R0 to set or reset the corresponding memory plane, the corresponding bit in this register must be set to 1. If the bit is set to 0, the corresponding set-reset bit in R0 is not enabled. The relationships are the same as those shown for R0 in Table 17-40.

Color Compare Register (R2) — The mode register (R5) must be in the read mode. If the data read from the memory planes compares with the corresponding bits in this register, a 1 will be placed in that position of the data bus. In essence, the four bits of this register are ANDed with the bits from corresponding memory planes. If they are equal (in agreement), the bit on the data bus will be set (1). If they do not compare, the bit on the data bus will be reset (0). Table 17-41 provides an example of this action when the color compare register is set to 3 (0011).

Table 17-41. Color Compare Register Example

D0	D1	D2	D3	D4	D5	D6	D7	EXPLANATION
1	1	1	1	1	1	1	1	The data in memory plane 0, which is compared with bit 0 of the color compare register. In this example, the bit is set to 1. Note that every bit agrees.
0	0	0	0	0	0	0	1	The data in memory plane 1, which is compared with bit 1 of the color compare register. In this example, the bit is set to 1. Note that only bit 7 agrees.
1	1	1	1	1	1	1	0	The data in memory plane 2, which is compared with bit 2 of the color compare register. In this example, the bit is set to 0. Note that only bit 7 agrees.
0	0	0	0	0	0	0	0	The data in memory plane 3, which is compared with bit 3, of the color compare register. In this example, the bit is set to 0. Note that every bit agrees.
0	0	0	0	0	0	0	1	The result of color compare action, which is placed on data bus. Bit 7 is set because the bits from all memory planes agree with the bits in the color compare register. The other bits are not set because in each instance a bit from at least one memory plane does not agree with a corresponding bit in the color compare register.

Data Rotate Register (R3) — This register is used to perform a rotate function on the data written by the CPU and allows the data in the CPU latches to be logically operated upon by the data being written. The value in the rotate count field represents the number of bits the CPU data will be rotated during CPU write cycles. Refer to Table 17-42 for a description of each bit in this register.

Table 17-42. Graphics Register R3

BIT	DESCRIPTION
0	Rotate Count 0.
1	Rotate Count 1.
2	Rotate Count 2.
3 and 4	When both bits are 0, the data will not be affected. When bit 3 is 1 and bit 4 is 0, a logical AND will be performed. When bit 3 is 0 and bit 4 is 1, a logical OR will be performed. When both bits are 1, a logical XOR will be performed.
5 and 6	These two bits are not used.

Read Map Select Register (R4) — The binary value in this register represents the memory plane number to be read by the CPU during a read operation. That is, if the binary value is 2, then memory plane 2 will be read.

If you write any value over 3, only the first two bits will be considered. Therefore, if you write a 7 to this register, the value will be interpreted to be 3; 5 will become 1, and so on.

NOTE: The value in this register does not affect the read operation performed through the color compare register (R2).

Mode Register (R5) — The mode register determines the type of operation this card performs: write mode, test mode, odd/even mode, and shift register mode. Table 17-43 describes the bits that define each mode.

Table 17-43. Modes of Operation

BITS	SETTING	DESCRIPTION
0,1		Write Mode. The following write operations work in conjunction with the function select options programmed in the data rotate register(R3).
	0,0	Write mode 0: All four memory planes are written with the CPU data rotated by the count in the rotate register (R3). If the set-reset register is enabled (R1) for any of the four memory lanes, the corresponding planes are written with the data stored in the set-reset register (R0).
	0,1	Write mode 1: All four memory planes are written with data that has been previously loaded into the CPU latches.
	1,0	Write mode 2: Memory planes 0 through 3 are filled with the value represented by data bits 0 through 3, respectively.
	1,1	This is an illegal write mode and nothing will happen.
2		Used in testing the IC.
	0	Normal operation.
	1	Test condition. The GRAPHICS, CHAIN, CDSEL0, and CDSEL1 outputs will be placed into a three-state condition for testing the IC. ATR0 through ATR3 are not placed in the three-state condition.
3		Read Modes
	0	The read map select register (R4) specifies the memory plane to be read by the CPU.
	1	The contents of the color compare register (R2) is used to logically read the contents of the four memory planes. See the discussion covering the color compare register for an example of this operation.

Table 17-43 (continued). Modes of Operation

BITS	SETTING	DESCRIPTION
4		Odd/Even addressing mode.
	0	Normal operation (sequential addressing).
	1	This selects the off/even addressing mode which is used when emulating the IBM Color Graphics Adapter modes.
5		Shift register mode. Output is sent through ATRO (memory plane 0) through ATR3 (memory plane 3).
	0	Normal operation — See Table 17-44 for the format. The data bits in memory planes 0 through 3 are represented as M0D0 through M0D7, M1D0 through M1D7, M2D0, through M2D7, and M3D0 through M3D7, respectively. The first two registers are identified as the Graphics A shift register and the second two registers are the Graphics B shift register.
	1	Shift operation — See Table 17-45 for the format. Bits M0D7 through M0D0, M1D7, through M1D0, M2D7 through M2D0, and M3D7 through M3D0 are shifted out with bit D7 first in all cases. The first two registers are identified as the Graphics A shift register and the second two registers are the Graphics B shift register.
6,7		Not used.

Table 17-44. Shift Register Mode Output Format, Bit 5 is 0

MSB								LSB	OUTPUT
M0D7	M0D6	M0D5	M0D4	M0D3	M0D2	M0D1	M0D0		ATRO
M1D7	M1D6	M1D5	M1D4	M1D3	M1D2	M1D1	M1D0		ATR1
M2D7	M2D6	M2D5	M2D4	M2D3	M2D2	M2D1	M2D0		ATR2
M3D7	M3D6	M3D5	M3D4	M3D3	M3D2	M3D1	M3D0		ATR3

Table 17-45. Shift Register Mode Output Format, Bit 5 is 1

MSB								LSB	OUTPUT
M1D0	M1D2	M1D4	M1D6	M0D0	M0D2	M0D4	M0D6		ATR0
M1D1	M1D3	M1D5	M1D7	M0D1	M0D3	M0D5	M0D7		ATR1
M3D0	M3D2	M3D4	M3D6	M2D0	M2D2	M2D4	M2D6		ATR2
M3D1	M3D3	M3D5	M3D7	M2D1	M2D3	M2D5	M2D7		ATR3

Miscellaneous Register (RG) — The miscellaneous register provides additional control for miscellaneous operations performed by the EGA chip set. Table 17-46 describes the operation and control provided by each bit.

Table 17-46. Miscellaneous Register

BITS	SETTING	DESCRIPTION
0		Select the Graphics Mode. This bit is available on the GRAPHICS pin of the 82C431.
	1	The character generator latches, which are located outside the EGA chip set, are disabled, enabling the graphics mode.
	0	Normal text mode.
1		Chain odd maps to even. This bit is available on the CHAIN pin of the 82C431. The function is external to the 82C431.
	1	Address bit 0 (A0) is replaced by a higher order address bit. If bit A0 is 0, memory planes 0 and 2 will be selected. If A1 is 1, memory planes 1 and 3 will be selected.
	0	Normal operation.

Table 17-46 (continued). Miscellaneous Register

BITS	SETTING	DESCRIPTION
2 and 3		These two bits map the address memory buffers into CPU address space. Bits 2 and 3 are available on pins CDSELO and CDSEL1 of the 82C431. The mapping action is external to the EGA chip set.
	$\begin{smallmatrix} 2 & 3 \\ \hline & 0 \end{smallmatrix}$	0 Map 128K at A000H. When this mapping is used no other video card may be actively installed in the system.
	0 1	Map 64K at A000H.
	1 0	Map 32K at B000H.
	1 1	Map 32K at B800H.
4 – 7		Not used.

Color Don't Care Register (R7) — The color don't care register works in conjunction with the color compare register (R2). Bits 0 through 3 correspond to memory planes 0 through 3. When any bit is set, the corresponding memory plane will be ignored (don't care) during the color compare operation. Refer to the description of R2 provided previously in this section. Bits 4 through 7 are not used in this register.

Bit Mask Register (R8) — The bit mask register controls access to all four memory planes simultaneously. Bits 0 through 7 correspond to bits 0 through 7 in each memory plane at the address selected.

If a bit is set to 0, the corresponding bit in each memory plane cannot be changed by the CPU. The data written to memory will be the same data that was read in the previous cycle, which is stored in an internal latch of the 82C431.

31 kHz Video

If a bit is set to 1, the corresponding bit in each memory plane can be manipulated by the CPU, including rotate, logical functions, set-reset, and no change.

To preserve data for use by the bit mask register, perform a read operation of the memory address. This will set the 82C431's internal latches.

82C431 Pinout

The 82C431 graphics controller is a 68-pin, very-large-scale integrated circuit. The pin definitions are described in Table 17-47 illustrated in Figure 17-4.

Table 17-47. 82C431 Graphics Controller Pin Functions

PIN NUMBER	SIGNAL NAME	DEFINITION
1	Vss	Ground.
2	M3D1	Memory plane 3, bit 1. Input/three-state output. Used to send data to and receive data from, the video memory.
3	M3D0	Memory plane 3, bit 0. Input/three-state output. Used to send data to and receive data from, the video memory.
4	A0	Address line 0. Input from the CPU address bus. Used to select the internal registers of the 82C431.
5	A1	Address line 1. Input from the CPU address bus. Used to select the internal registers of the 82C431.
6	A2	Address line 2. Input from the CPU address bus. Used to select the internal registers of the 82C431.

Table 17-47 (continued). 82C431 Graphics Controller Pin Functions

PIN NUMBER	SIGNAL NAME	DEFINITION
7	$\overline{\text{IOW}}$	Input/Output Write. Active low- input from the CPU. Used to load the internal registers of the 82C431
8,9	MOD7, MOD6	Memory plane 0, bits 7 and 6. Input/three-state output. Used to send data to and receive data from, the video memory.
10	Vss	Ground.
11 – 16	MOD5- MOD0	Memory plane 0, bits 5 through 0. Input/three-state output. Used to send data to and receive data from, the video memory.
17	Vdd	+ 5 VDC supply.
18	Vss	Ground.
19	DOTCLK	Dot clock. Input used to shift data in shift register.
20	$\text{S}/\overline{\text{L}}$	Shift/Load. An input signal generated by the 82C432. When low, the display memory data will be loaded into the shift register. When high, the data will be shifted in the shift register by one bit.
21	$\overline{\text{CRTL}}$	CRT latch. Active low input from the 82C432. When active, the display data will be loaded into the 82C431 on the positive-going edge. Loaded data is transferred into the shift register when this signal is low.
22	$\overline{\text{CPUL}}$	CPU latch. Active low input from the 82C432. When active the video memory data will be read and loaded into the 82C431. The loaded data is transferred onto the data bus on the positive-going edge of this signal and when the MRD signal is active.

31 kHz Video

Table 17-47 (continued). 82C431 Graphics Controller Pin Functions

PIN NUMBER	SIGNAL NAME	DEFINITION
23	CDSEL1	CD select 1. An output of the miscellaneous register (see "R6 — Miscellaneous Register"). Used with CDSELO to control the mapping of the memory planes.
24	CDSELO	CD select 0. An output of the miscellaneous register (see "R6 — Miscellaneous Register"). Used with CDSEL1 to control the mapping of the memory planes.
25	$\overline{\text{WE}}$	Write enable. An active low input from the CPU. Used to send display memory on the memory data buses. When high, the data buses are placed in the high impedance mode.
26 – 29	MID7- MID4	Memory plane 1, bits 7 through 4. Input/three-state output. Used to send data to and receive data from, the video memory.
30	M1D2	Memory plane 1, bit 2. Input/three-state output. Used to send data to and receive data from, the video memory.
31	ATR3	Attribute 3. Used to transfer the output of the memory planes to the attributes controller (82C433).
32	MID2	Memory plane 1, bit 2. Input/three-state output. Used to send data to and receive data from, the video memory.
33	ATR2	Attribute 2. Used to transfer the output of the memory planes to the attributes controller (82C433).
34	Vdd	+ 5 VDC supply.
35	Vss	Ground.

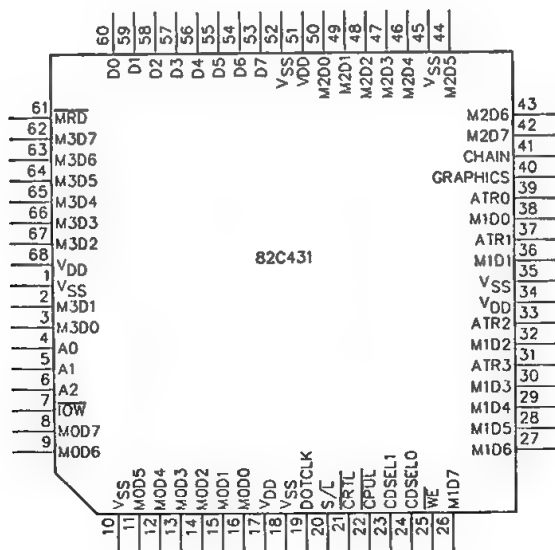
Table 17-47 (continued). 82C431 Graphics Controller Pin Functions

PIN NUMBER	SIGNAL NAME	DEFINITION
36	M1D1	Memory plane 1, bit 1. Input/three-state output. Used to send data to and receive data from, the video memory.
37	ATR1	Attribute 1. Used to transfer the output of the memory planes to the attributes controller (82C433).
38	M1D0	Memory plane 1, bit 0. Input/three-state output. Used to send data to and receive data from, the video memory.
39	ATR0	Attribute 0. Used to transfer the output of the memory planes to the attributes controller (82C433).
40	GRAPHICS	Graphics mode. An output of the miscellaneous register (see "R6 — Miscellaneous Register"). Used with CHAIN to multiplex the address buses in the 82C434.
41	CHAIN	Chain odd maps to even. An output of the miscellaneous register (see "R6 — Miscellaneous Register"). Used with GRAPHICS to multiplex the buses in the 82C434.
42 – 44	M2D7- M2D5	Memory plane 2, bits 7 through 5. Input/three-state output. Used to send data to and receive data from, the video memory.
45	Vss	Ground.
46 – 50	M2D4- M2D0	Memory plane 2, bits 4 through 0. Input/three-state M2D0 output. Used to send data to and receive data from, the video memory.
51	Vdd	+ 5 VDC supply.

31 kHz Video

Table 17-47 (continued). 82C431 Graphics Controller Pin Functions

PIN NUMBER	SIGNAL NAME	DEFINITION
52	Vss	Ground.
53–60	D7–D0	CPU data bus, bits 7 through 0. A bidirectional, three-state signal. Used to load data into the 82C431 and to send data to and receive data from, the video memory.
61	MRD	Memory read. An active low input from the CPU. Used to read video memory data in conjunction with the positive-going leading edge of the CPUL input.
62–67	M3D7–M3D2	Memory plane 3, bits 7 through 2. Input/three-state output. Used to send data to and receive data from, the video memory.
68	Vdd	+ 5 VDC supply.

**Figure 17-4. 82C431 Graphics Controller Pinout**

82C432 Sequencer

The 82C432 is the timing generator for the EGA chip set. It provides the memory timing signals, the character clock, and the dot clock. In addition, the sequencer allows the CPU to access video memory during active display intervals by inserting dedicated CPU memory cycles into the system, which provides different screen resolutions. A configurable plane mask register protects the entire memory from being altered.

Table 17-48 summarizes the internal registers of the 82C432 sequencer.

Table 17-48. 82C432 Sequencer Register Summary

PORT ADDRESS	REGISTER NUMBER	REGISTER NAME
3C4	—	Address register (00 – 04)
3C5	R0	00 — reset register
3C5	R1	01 — clocking mode register
3C5	R2	02 — plane mask register
3C5	R3	03 — character map select register
3C5	R4	04 — memory mode register

Address Register — The address register is used to point to the other registers in the 82C432 and route input and output to the indicated register through port 3C5H. The three least significant bits determine the register selected. Table 17-49 defines the address of each register.

Table 17-49. Sequencer Addressing

BIT 2	BIT 1	BIT 0	VALUE	REGISTER
0	0	0	00	R0 — reset register
0	0	1	01	R1 — clocking mode register
0	1	0	02	R2 — plane mask register
0	1	1	03	R3 — character map select register
1	0	0	04	R4 — memory mode register

31 kHz Video

The partial program in Listing 7-2 illustrates a method that you may use to place a value (previously loaded into the CPU register BL) into the character map select register of the 82C432 sequencer. Loading data into other registers follows a similar pattern.

Listing 17-2. 82C432 Programming Example

```
MOV DX, 3C4H    ;select the register
MOV AL, 03      ;port address of the address register
OUT DX, AL      ;select the register for the data
                ;send AL to port defined by DX
                ;send the data
MOV DX, 3C5H    ;port address for internal register
MOV AL, BL      ;place value previously loaded in BL into AL
OUT DX, AL      ;send value to register
```

Reset Register (R0) — The reset register controls the method of resetting the sequencer for timing purposes.

If bit 0 is clear (set to 0), the sequencer will clear in an asynchronous manner and halt. This method can cause data to be lost in the video memory. While halted, all outputs will be placed in a high impedance state. If bit 1 is set (to 1), the sequencer will run unless a synchronous reset has been implemented by bit 1 of this register.

If bit 1 is clear (set to 0), the sequencer will clear in a synchronous manner and halt. This method preserves video memory contents and should be used before changing the clocking mode register (R1). While halted, all outputs will be placed in a high impedance state. If bit 1 is set (to 1), the sequencer will run unless an asynchronous reset has been implemented by bit 0 of this register.

Bits 2 through 7 are not used. Both bit 0 and 1 must be set for the sequencer to run.

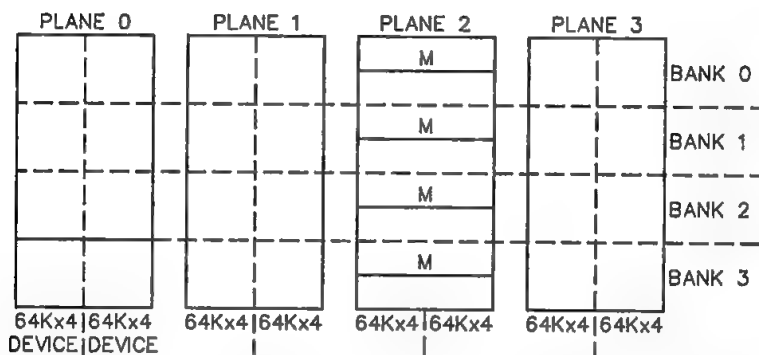
Clocking Mode Register (R1) — This register controls the timing signals for the various video modes. Table 17-50 describes the action of each bit in the register.

Table 17-50. Clocking Mode Register Functions

BIT	DESCRIPTION
0	<p>8- or 9-pixel dot clock selection.</p> <p>If bit 0 is clear (set to 0), the sequencer will generate character clocks that are 9 pixels wide. The only mode that uses this mode is the monochrome video mode 7, which results in a screen resolution of 720×350 pixels. If bit 0 is set (to 1), the sequencer will generate character clocks that are 8 pixels wide. This mode is used by all video modes except monochrome video mode 7.</p>
1	<p>Bandwidth selection.</p> <p>If bit 1 is clear, CRT memory cycles occur every 4 out of 5 video memory cycles. All high-resolution modes of 640 or 720 pixels must provide the CRT controller with this bandwidth, which is required to refresh to video image. If bit 1 is set, CRT memory cycles occur every 2 out of 5 video memory cycles. Medium resolution modes of 320 pixels do not require as much access to video memory as the higher-resolution modes. Therefore, the CPU can have more time to access video memory.</p>
2	<p>Shift Load.</p> <p>If bit 2 is clear, the display shift registers in the 82C431 will be loaded every character clock. If bit 2 is set, the display shift registers in the 82C431 will be loaded every other character clock. This bit is set only when bit-mapped graphics are being displayed; 16-bits are fetched every memory cycle and chained together in the shift registers.</p>
3	<p>Dot Clock.</p> <p>If bit 3 is clear, the dot clock follows the same frequency as the master clock input. If bit 3 is set, the dot clock is one-half the frequency of the master clock input. This is the setting used for 320×200-pixel resolution modes.</p>
4-7	Not used.

31 kHz Video

Plane Mask Register (R2) — This register is sometimes referred to as the map mask register. Refer to Figure 17-5. The maps correspond to the upper 8K of each of the four 16K banks in memory plane 2. Bits 0 through 3 correspond to memory banks 0 through 3 of memory plane 2. These bits disable the corresponding RAS0 through RAS3 signals during CPU write operations. Therefore, if this register is loaded with 0FH, the CPU can perform a 32-bit write operation in one memory cycle, substantially reducing the over-head required during a display update cycle in the graphics modes.



M=MEMORY MAPS IN PLANE 2. EACH MAP CORRESPONDS TO 8 KBYTES OF MEMORY.

BANKS 0,1,2,3 CORRESPOND TO 16 KBYTES OF MEMORY SEGMENTS IN EACH OF THE FOUR PLANES.

IN THE ALPHANUMERIC MODE: PLANE 0 HAS ADDRESS DATA
 PLANE 1 HAS ATTRIBUTE DATA
 PLANE 2 HAS PIXEL (CHARACTER) DATA
 PLANE 3 NOT USED

Figure 17-5. Video Memory Configuration

When an odd/even mode is selected by clearing bit 2 of the memory mode register in the 82C431, planes 0 and 1 and planes 2 and 3 will have the same plane mask value.

Bits 4 through 7 are not used.

Character Map Select Register (R3) — The character map select register only affects the B-address bus signals BCAS, BAV14, and BAV15. Bits 0 and 1 select the map used to generate alphanumeric characters when attribute bit 3 is clear (0). Refer to Table 17-51. Bits 2 and 3 select the map used to generate alphanumeric characters when attribute bit 3 is set (1). Refer to Table 17-52.

In alphanumeric modes, bit 3 of the attribute byte normally turns the foreground intensity on or off. The bit may be redefined as a switch between character sets. This function is enabled whenever there is a difference between the values of the character map select A and the character map select B bits. If the values are equal, the character select function is disabled.

If bit 1 of the memory mode register (R4) is clear, there is 256K of video memory and this function can be enabled. Four character sets are supported.

If bit 1 is set, then the video card has only 64K of video memory and bank 0 is always selected.

The asynchronous reset operation clears the character map select register (resets it to 0). To avoid altering the contents of memory, the asynchronous reset operation should be used only during a system reset.

Table 17-51. Character Map Select A (Bits 0 and 1)

BIT 1	BIT 0	MAP	LOCATION
0	0	0	First 8K of plane 2, bank 0
0	1	1	First 8K of plane 2, bank 1
1	0	2	First 8K of plane 2, bank 2
1	1	3	First 8K of plane 2, bank 3

31 kHz Video

Table 17-52. Character Map Select B (Bits 2 and 3)

BIT 1	BIT 0	MAP	LOCATION
0	0	0	First 8K of plane 2, bank 0
0	1	1	First 8K of plane 2, bank 1
1	0	2	First 8K of plane 2, bank 2
1	1	3	First 8K of plane 2, bank 3

Memory Mode Register (R4) — The memory mode register contents and actions are described in Table 17-53.

Table 17-53. Memory Mode Register

BIT	DESCRIPTION
0	Alphanumeric mode. When this bit is clear, a non-alphanumeric mode is active. BAV14 = AAV14 and BAV15 = AAV15. When this bit is set, the alphanumeric mode is active and the character map select register (R3) is enabled.
1	Extended memory. When this bit is clear, video memory consists of 1 bank of 16 kilobit by 4 devices, providing a total of 64K of video memory. Address bits AAV14, AAV15, BAV14, and BAV15 are disabled. When this bit is set, video memory consists of 1 bank of 64 kilobit by 4 devices, providing a total of 256K of video memory. The high order address bits AAV14, AAV15, BAV14, and BAV15 are enabled.
2	Odd/Even. When this bit is clear, even CPU addresses are sent to maps 0 and 2 (RAS0 and RAS2 are enabled). Odd CPU addresses are sent to maps 1 and 3 (RAS1 and RAS3 are enabled). When this bit is set, CPU addresses are sent sequentially to the bit maps (all RAS signals are enabled).
3–7	Not used.

82C432 Pinout

The 82C432 sequencer is a 40-pin, large-scale integrated circuit. The pin definitions are described in Table 17-54 and illustrated in Figure 17-6.

Table 17-54. 82C432 Sequencer Pin Functions

PIN NUMBER	SIGNAL NAME	DEFINITION
1	Vss	Ground.
2	CLKIN	Clock input. Input signal from the system that establishes all other timing signals. Referred to as the master clock signal.
3	CRA15	CRT address bit 15. Input from the 82C434 used as a high order address bit for the CRT display read operation.
4	CRA14	CRT address bit 14. Input from the 82C434 used as a high order address bit for the CRT display read operation.
5	CPA15	CPU address bit 15. Input from the decoded CPU address bus signals A13 – A19. Used for CPU read/write operations. During memory read and write cycles, AAV15 = BAV15 = CPA15.
6	CPA14	CPU address bit 14. Input from the CPU used for CPU read/write operations. During memory read and write cycles, AAV14 = BAV14 = CPA14.
7	$\overline{\text{ACAS}}$	Column access strobe A. Active low signal. Output strobe signal for memory position A.
8	AAV15	Video address 15 A. Used as the high order bit for memory map A. During memory read and write AAV15 = BAV15 = CPA15.

31 kHz Video

Table 17-54 (continued). 82C432 Sequencer Pin Functions

PIN NUMBER	SIGNAL NAME	DEFINITION
9	AAV14	Video address 14 A. Used as a high order bit for memory map A. During memory read and write AAV14 = BAV14 = CPA14.
10	$\overline{\text{BCAS}}$	Column access strobe B. Active low signal. Output strobe signal for memory position B.
11	BAV15	Video address 15 B. Used as the high order bit for memory map B. During memory read and write AAV15 = BAV15 = CPA15.
12	BAV14	Video address 14 B. Used as a high order bit for memory map B. During memory read and write AAV14 = BAV14 = CPA14.
13	$\overline{\text{CRTL}}$	CRT latch. This active low output is used to latch the memory data into the 82C431 and the 82C433 for CRT display read cycles.
14	$\overline{\text{CPUL}}$	CPU latch.
15	$\overline{\text{IOW}}$	Input/output write. This active low input signal from the system bus is used to load the various registers in the 82C432.
16	$\overline{\text{WE}}$	Write enable. This active low output allows the CPU to write to video memory.
17	SYNC	Sync. This input from the 82C434 is used to disable the AS/L signal during non-active display periods.
18	MUX	Multiplexer. This output is used to multiplex the RAS and CAS signals.
19	DOTCLK	Dot Clock. This output clock is used by the 82C431 and 82C433.
20	Vss	Ground.

Table 17-54 (continued). 82C432 Sequencer Pin Functions

PIN NUMBER	SIGNAL NAME	DEFINITION
21	$\overline{\text{RAS0}}$	Row access strobe plane 0. This active low signal is used to access memory plane 0.
22	$\overline{\text{RAS1}}$	Row access strobe plane 1. This active low signal is used to access memory plane 1.
23	$\overline{\text{RAS2}}$	Row access strobe plane 2. This active low signal is used to access memory plane 2.
24	$\overline{\text{RAS3}}$	Row access strobe plane 3. This active low signal is used to access memory plane 3.
25	$\text{CRT}/\overline{\text{CPU}}$	CRT/CPU control signal. This output is used by the 82C434 CRT controller to enable either the CRT read address or the CPU read/write address.
26	CCLK	Character clock. Output clock signal used by the 82C434.
27	$\text{AS}/\overline{\text{L}}$	Attribute shift/load. Output signal used to load alphanumeric video memory into the 82C433.
28	$\text{S}/\overline{\text{L}}$	Shift/Load. This output is used to load video memory graphics data into the 82C431.
29	$\overline{\text{CRDY}}$	Clock ready. This output is similar to the RDY output. The difference between the two signals is the way they go inactive (high). RDY goes inactive and stays inactive (high). CRDY goes inactive, but stays high for one dot period and then goes into a high-impedance state.
30	$\overline{\text{RDY}}$	Ready. This active low output is used to indicate that the video memory read or write operation is completed.

31 kHz Video

Table 17-54 (continued). 82C432 Sequencer Pin functions

PIN NUMBER	SIGNAL NAME	DEFINITION
31	MEMOPT	Memory option. The input control signal is used to select 256 K video memory option when low or 64K video memory option when high.
32 – 35	DO – D3	Data bits 0 through 3. Used to load the sequencer register.
36	A0	Address 0. Input signal from CPU address bus. Used to select between the address register and the register pointed to by the address register. It is also used to write into even/odd memory planes.
37	CG	Character generator. Input from video memory (attribute bit 3) used to select between character map A and character map B.
38	$\overline{\text{MEMW}}$	Memory write. This active low input from the system is used to write data into the video memory.
39	$\overline{\text{MEMR}}$	Memory read. This active low input from the system is used to read data from video memory.
40	Vdd	+ 5 VDC supply.

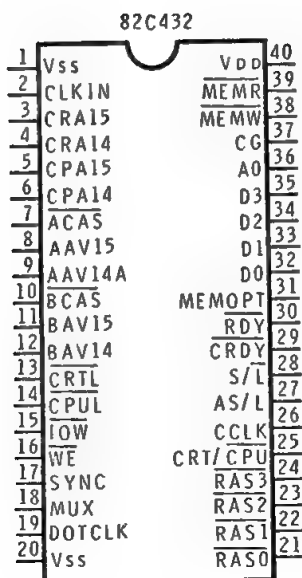


Figure 17-6. 82C432 Sequencer Pinout

82C433 Attributes Controller

The 82C433 is the attributes controller of the EGA chip set. It provides a palette of 16 colors that may be specified from any of 64 possible colors. This chip also controls the blinking and underline functions. It takes data from the video memory and formats it for display on the screen. And it provides horizontal pixel panning capability to the bit-mapped and alphanumeric modes.

The EGA chip set supports six video output signals supplied by the 82C433: red, green, and blue are identical to other RGB video signals; while secondary red, secondary green/intensity, and secondary blue/monochrome are unique to the EGA design. The monochrome signal is identical to the composite monochrome signal of Z-100 PC color video cards, but shares its output pin with the secondary blue signal. The mode of operation determines which signal these two output pins carry. Dual frequency enhanced graphics monitors are capable of decoding the signals and correctly displaying the information.

31 kHz Video

Besides the six color video signals, a four-bit status port provides signals to the rest of the system. Table 17-55 describes the output of the status port.

Table 17-55. 82C433 Status Port

BIT	DESCRIPTION
0	DE — Display enable. This active low output is the inverted display enable output from the 82C434 CRT controller.
1 and 2	Not used.
3	VRTC — Vertical retrace. This reflects the vertical retrace output of the 82C434 CRT controller.
4 and 5	These are two of the six color outputs of the palette registers. See "R0 – RF – Palette Registers."
6 and 7	Not used.

Table 17-56 summarizes the twenty internal registers in the 82C433 Attributes controller.

Table 17-56. 82C433 Register Summary

PORT ADDRESS	REGISTER NUMBER	REGISTER NAME
3C0	—	Address register (00H – 13H)
3C0	R0 – RF	00H – 0FH – palette registers 1 – 16
3C0	R10	10H – mode control register
3C0	R11	11H – overscan color register
3C0	R12	12H – color plane enable register
3C0	R13	13H – horizontal pixel panning register

Address Register — The address register in the 82C433 functions slightly differently than the address registers in the other chips of the EGA chip set. Bits 0 through 4 of the address register are used to point to the other registers and route input and output to the indicated register through port 3C0H. The five least significant bits determine the register selected. Table 17-57 defines the address of each register.

Bit 5 is the read/write select flag for the palette registers. If this bit is set, the color palette registers may be accessed for CRT operation. If this bit is clear, the registers may be loaded with data.

Table 17-57. Attributes Controller Addressing

BIT 4	BIT 3	BIT 2	BIT 1	BIT 0	VALUE	REGISTER
0	0	0	0	0	00	R0 — palette register 1
0	0	0	0	1	01	R1 — palette register 2
0	0	0	1	0	02	R2 — palette register 3
0	0	0	1	1	03	R3 — palette register 4
0	0	1	0	0	04	R4 — palette register 5
0	0	1	0	1	05	R5 — palette register 6
0	0	1	1	0	06	R6 — palette register 7
0	0	1	1	1	07	R7 — palette register 8
0	1	0	0	0	08	R8 — palette register 9
0	1	0	0	1	09	R9 — palette register 10
0	1	0	1	0	0A	RA — palette register 11
0	1	0	1	1	0B	RB — palette register 12
0	1	1	0	0	0C	RC — palette register 13
0	1	1	0	1	0D	RD — palette register 14
0	1	1	1	0	0E	RE — palette register 15
0	1	1	1	1	0F	RF — palette register 16
1	0	0	0	0	10	R10 — mode control register
1	0	0	0	1	11	R11 — overscan color register
1	0	0	1	0	12	R12 — color plane enable register
1	0	0	1	1	13	R13 — horizontal pixel panning register

31 kHz Video

Loading a register is also different than the other chips of the EGA chip set. An internal flip-flop in the chip selects the address or data registers accessed through port 3C0H. To force the flip-flop to select the address register, send an IN command to the port. Then use an OUT command to place the register number in the address register. This action also toggles the flip-flop so that the next OUT command will access the selected data register. The partial program in Listing 17-3 illustrates this method to place a value (previously loaded into the CPU register BL) into the fourth palette register (R3) of the 82C433 attributes controller. Loading data into the other registers follows a similar pattern.

Listing 17-3. 82C433 Programming Example

	;select the register
MOV DX, 3C0H	;port address of attributes controller
IN DX, AL	;toggle flip-flop to select address register
MOV AL, 03	;select the register for the data
OUT DX, AL	;send AL to port defined by DX
	;send the data
MOV AL, BL	;place value previously loaded in BL into AL
OUT DX, AL	;send value to register

Palette Registers (R0 – RF) — The sixteen palette registers allow a dynamic mapping between the text attribute or graphic color input and the display color on the CRT screen. The function of the sixteen registers is identical, in that it allows the user to specify any one of 64 possible colors. Since each register may contain any 6 bit code, it is possible to display any 16-color combination on the screen. Table 17-58 describes the output from each bit in the palette registers.

The color palette registers should be loaded only during vertical retrace time to avoid producing glitches on the screen. An interrupt is generated by the CRT controller (CRTINT) to indicate the end of the vertical display enable signal.

Table 17-58. Palette and Overscan Registers Video Output

BIT	COLOR OUTPUT
0	Blue
1	Green
2	Red
3	Secondary blue or composite monochrome
4	Secondary green or intensity
5	Secondary red
6 and 7	Not used

Mode Control Register (R10) — The mode control register provides control of certain attributes in specific modes. The attributes and modes are defined in Table 17-59.

Table 17-59. Mode Control Register Functions

BIT	DESCRIPTION
0	This bit selects between the graphics and alphanumeric mode. If the bit is clear, the alphanumeric mode is selected. If the bit is set, the graphics mode is elected.
1	This bit selects between color and monochrome display attributes. If the bit is clear, the color display attributes are selected. If the bit is set, the monochrome attributes are selected.
2	This bit provides compatibility with monochrome card output in video mode 7. If this bit is clear, the ninth pixel in a scan line will display the background color. If this bit is set, the special line graphics character codes for the monochrome card are enabled. The line graphics characters are codes C0H through DFH. The CRT controller generates the line graphics signal, LG, whenever one of these codes is to be displayed. The 82C433 monitors LG whenever bit 2 is set, and when it detects the LG signal to be asserted, it forces the ninth pixel of line graphics character scan line to be identical to the eight pixel of that same scan line.

31 kHz Video

Table 17-59 (continued). Mode Control Register Functions

BIT	DESCRIPTION
3	This bit enables the blink attribute or selects the background intensity. If this bit is clear, the background intensity is selected for monochrome card and Z-100 PC color video card compatibility. If this bit is set, the blink attribute is enabled in alphanumeric and graphics modes. The blink counter operates by dividing the VRTC input by 32. In the alphanumeric modes, blink is usually used in monochrome display modes only where the blink allows the selected character to be displayed for 16 frames and then inverted for 16 frames, allowing two different colors to be displayed. The character at the cursor is blinked at the same rate, but the cursor itself is blinked at the rate of 8 frames on and 8 frames off.
4 - 7	Not used.

Overscan Color Register (R11) — Bits 0 through 5 of the overscan color register define the border color (overscan color) to be displayed on the screen. The border color is displayed when both BLANK and DE are inactive. The 6 bits are defined in the previous table, 17-58. Bits 6 and 7 are not used.

Color Plane Enable Register (R12) — The color plane enable register controls the enabling of the memory planes and selects two of the six colors to run special diagnostics.

Bits 0 through 3, when set, enable the corresponding memory plane 0 through 3.

Bits 4 and 5 select two of the six color outputs as described Table 17-60. These two bits are reflected in bits 2 and 3 of the status register. Also, when bit 4 is set, the color monitor outputs, R, G, B, RS, GS/I, and BS/V will be placed in a high-impedance state and the cursor blink counter will be cleared. Bit 4 must be cleared (set to 0) before the blink counter will continue counting.

Table 17-60. Color Plane Enable Register Bits 4 and 5

BIT 5	BIT 4	DESCRIPTION
0	0	Bit 4 carries the blue output. Bit 5 carries the red output.
0	1	Bit 4 carries the green output. Bit 5 carries the secondary blue output.
1	0	Bit 4 carries the secondary red output.
1	1	Bits 4 and 5 are not used.

Horizontal Pixel Panning Register (R13) — The horizontal pixel panning register shifts the display area horizontally to the left. Panning is available in both the alphanumeric and bit-mapped modes.

The number of pixels the display area is panned is determined by the value of bits 0 through 3 of this register. The maximum number of pixels is one character width (9 pixels for monochrome mode 7 and 8 pixels for all others).

Although the start address register of CRT controller specifies the address in video memory of the upper left-hand corner of the screen, pixel panning makes it possible to move the display a fraction of a character width to the left. Bits 4 through 7 are not used.

82C433 Pinout

The 82C433 attributes controller is a 40-pin, large-scale integrated circuit. The pin definitions are described in Table 17-61 and illustrated in Figure 17-7.

31 kHz Video

Table 17-61. 82C433 Attributes Controller Pin Functions

PIN NUMBER	SIGNAL NAME	DEFINITION
1	D0	Data bit 0. This bidirectional path provides the display enable (DE) status output signal and a data bus input signal.
2	D1	Data bit 1. This is a data bus input signal.
3	D2	Data bit 2. This is a data bus input signal.
4	D3	Data bit 3. This bidirectional path provides the vertical retrace (VRTC) output signal and data bus input signal.
5	D4	Data bit 4. This bidirectional path provides the color palette bit 0 output signal and data bus input signal.
6	D5	Data bit 5. This bidirectional path provides the color palette bit 1 output signal and data bus input signal.
7	$\overline{\text{CRTL}}$	CRT latch. This active low input from the 82C432 is used to load video memory data into the 82C433.
8	$\text{S}/\overline{\text{L}}$	Shift/load. This input signal is generated by the 82C432 as AS/L. It is used to load video memory data into the 82C43.
9	$\overline{\text{IOR}}$	Input/Output read. This active low input signal has the dual purpose of reading the status, as well as clearing the address/data flip-flop.
10	$\overline{\text{IOW}}$	Input/output write. This active low input signal loads either the address register, one of the 16 color palette registers, or one of the 4 control registers.
11	ATR7	Attribute 7. This high order attribute bit input signal comes from memory plane 1.

Table 17-61 (continued). 82C433 Attributes Controller Pin Functions

PIN NUMBER	SIGNAL NAME	DEFINITION
12	ATR6	Attribute 6. This high order attribute bit input signal comes from memory plane 1.
13	ATR5	Attribute 5. This high order attribute bit input signal comes from memory plane 1.
14	ATR4	Attribute 4. This high order attribute bit input signal comes from memory plane 1.
15	ATR3	Attribute 3. This low order attribute bit input signal from 82C431 could be either graphics data generated by the 82C431 or alphanumeric data multiplexed by the 82C431 along with graphics data.
16	ATR2	Attribute 2. This low order attribute bit input signal from 82C431 could be either graphics data generated by the 82C431 or alphanumeric data multiplexed by the 82C431 along with graphics data.
17	ATR1	Attribute 1. This low order attribute bit input-signal from 82C431 could be either graphics data generated by the 82C431 or alphanumeric data multiplexed by the 82C431 along with graphics data.
18	ATRO	Attribute 0. This low order attribute bit input signal from 82C431 could be either graphics data generated by the 82C431 or alphanumeric data multiplexed by the 82C431 along with graphics data.
19	DOTCLK	Dot Clock. This input signal comes from the 82C432.
20	Vss	Ground.

31 kHz Video

Table 17-61 (continued). 82C433 Attributes Controller Pin Functions

PIN NUMBER	SIGNAL NAME	DEFINITION
21	R	Red. This is the red output signal that drives the attached monitor.
22	G	Green. This is the green output signal that drives the attached monitor.
23	B	Blue. This is the blue output signal that drives the attached monitor.
24	RS	Secondary Red. This is the secondary red output signal that drives the attached monitor.
25	GS/I	Secondary green/intensity. This is the secondary green/intensity output signal that drives the attached monitor.
26	BS/V	Secondary blue/monochrome video. This is the secondary blue/monochrome output signal that drives the attached monitor.
27	BLANK	Blank. This input from the 82C434 CRT controller is low during the time the monitor is displaying data.
28	VRTC	Vertical retrace. This input from the 82C434 is low during the time the monitor is displaying data.
29	DE	Display enable. This input from the 82C434 is used for enabling the display area on the screen. When both DE and BLANK are inactive, the boarder area is being displayed. It is output through D0 as a status bit.

Table 17-61 (continued). 82C433 Attributes Controller Pin Functions

PIN NUMBER	SIGNAL NAME	DEFINITION
30	CURSOR	Cursor. This signal from the 82C434 provides two functions: 1. During DE active time, it indicates a valid cursor position. 2. If CURSOR is active during the high-to-low edge of BLANK, it indicates that the contents of the CRT controller underline location register (R14) are equal to the line count (that is, equal to the contents of the raster counter).
31	$\overline{\text{LG}}$	Line graphics. This active low input from the 82C434 is a logical NAND of MOD5, MOD6, and MOD7 (bits 5, 6, and 7 of memory plane 0).
32	CC0	Condition code 0. This input, along with the other condition codes, is loaded into a parallel-to-serial shift register. A low output from this register enables the low-order attribute bits (ATR0-ATR3), and a high output enables the high-order attribute bits (ATR4-ATR7), for color palette addressing.
33	CC1	Condition code 1. This input, along with the other condition codes, is loaded into a parallel-to-serial shift register. A low output from this register enables the low-order attribute bits (ATR0-ATR3), and a high output enables the high-order attribute bits (ATR4-ATR7), for color palette addressing.
34	CC2	Condition code 2. This input, along with the other condition codes, is loaded into a parallel-to-serial shift register. A low output from this register enables the low-order attribute bits (ATR0-ATR3), and a high output enables the high-order attribute bits (ATR4-ATR7), for color palette addressing.

Table 17-61 (continued). 82C433 Attributes Controller Pin Functions

PIN NUMBER	SIGNAL NAME	DEFINITION
35	CC3	Condition code 3. This input, along with the other condition codes, is loaded into a parallel-to-serial shift register. A low output from this register enables the low-order attribute bits (ATR0-ATR3), and a high output enables the high-order attribute bits (ATR4-ATR7), for color palette addressing.
36	CC4	Condition code 4. This input, along with the other condition codes, is loaded into a parallel-to-serial shift register. A low output from this register enables the low-order attribute bits ATR0-ATR3), and a high output enables the high-order attribute bits (ATR4-ATR7), for color palette addressing.
37	CC5	Condition code 5. This input, along with the other condition codes, is loaded into a parallel-to-serial shift register. A low output from this register enables the low-order attribute bits (ATR0-ATR3), and a high output enables the high-order attribute bits (ATR4-ATR7), for color palette addressing.
38	CC6	Condition code 6. This input, along with the other condition codes, is loaded into parallel-to-serial shift register. A low output from this register enables the low-order attribute bits (ATR0-ATR3), and a high output enables the high-order attribute bits (ATR4-ATR7), for color palette addressing.
39	CC7	Condition code 7. This input, along with the other condition codes, is loaded into a parallel-to-serial shift register. A low output from this register enables the low-order attribute bits ATR0-ATR3), and a high output enables the high-order attribute bits (ATR4-ATR7), for color palette addressing.
40	Vdd	+ 5 VDC supply

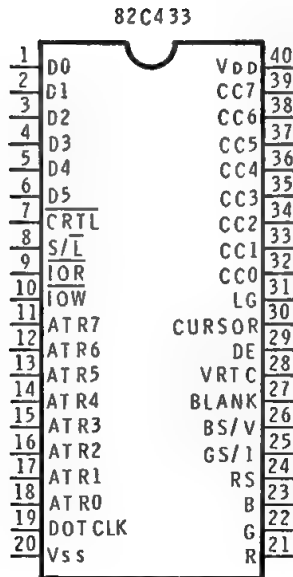


Figure 17-7. 82C433 Attributes Controller Pinout

82C434 CRT Controller

The CRT controller provides all timing and control signals for the video display. Figure 17-8 illustrates how the contents of the CRT controller's registers control the screen. The horizontal blanking signal is controlled by the contents of R2 and R3. The contents of R15 and R16 determine the location and length of the vertical blanking signal, BLANK, which is the result of ORing the HBLANK and VBLANK signals. The horizontal sync start location and width are controlled by the contents of R4 and R5, while the vertical sync start location and width are controlled by R10 and R11. R0 controls the total number of characters in the horizontal scan interval, including retrace time. The total number of scan lines on one page are determined by the vertical total register R6. R1 and R12 define the effective display area by specifying the horizontal and vertical display enable positions.

31 kHz Video

The CRT controller is also capable of split screen operation as illustrated in the Figure. The two screens, identified in the illustration as screen 1 and screen 2, are created by setting the memory address register RC and RD and the line compare register R18. RC and RD specify the memory address of the first pixel displayed on the screen. By using the line compare register R18, the internal start of the scan line counter is cleared when the vertical counter reaches the value equal to the contents of R18. This causes screen 2 to start at address location 0000H in video memory. Thus, normal scrolling activity is limited to screen 1, leaving screen 2 alone.

Finally, the CRT controller generates an interrupt at the end of a vertical display, which is controlled by the vertical display end register. The interrupt is used by the CPU to update the EGA card during the vertical blanking interval.

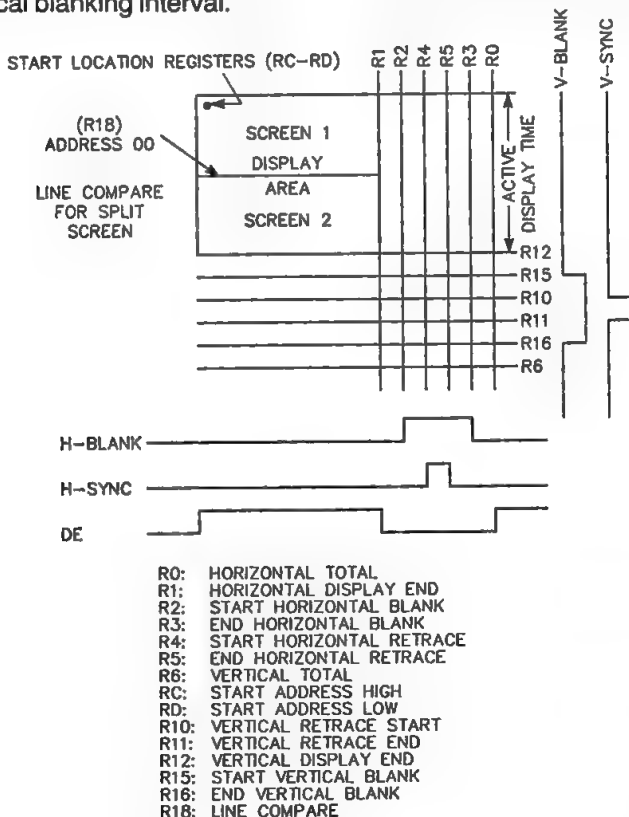


Figure 17-8. Display Timing

The CRT controller is capable of "soft scrolling." Refer to Figure 17-9. As previously described, RC and RD define the video memory address of the first pixel that is displayed on the screen. In the alphanumeric mode, this is the memory address of the first character in the first row. The present scan register R8 defines the starting scan row on the screen where the information is actually started. During the vertical retrace period, the starting address is latched by the CRT controller and R8 is loaded into the scan row counter. Updating the starting address registers and R8, creates a smooth vertical scrolling effect.

For graphics modes, R8 should be set to 0. Smooth vertical scrolling can be controlled by updating the starting address registers only. The figure also illustrates how subsequent addresses are computed for a given scan line.

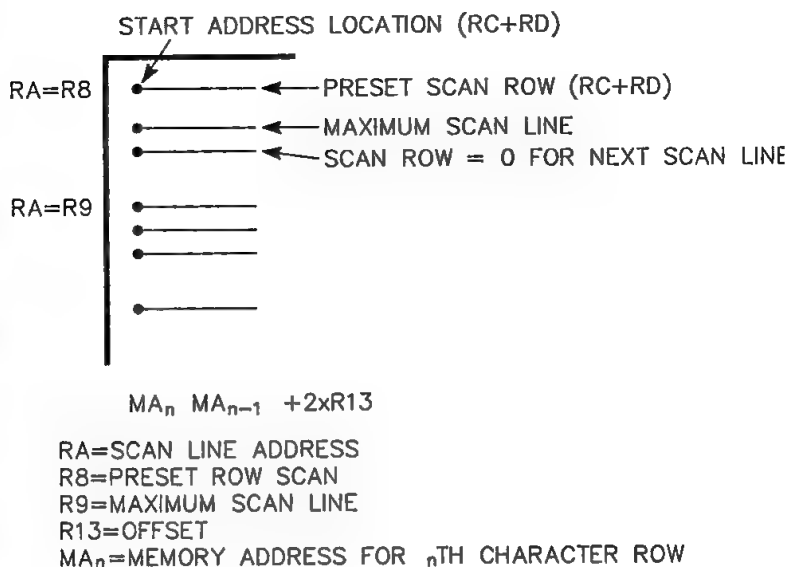


Figure 17-9. Soft Scrolling

31 kHz Video

The character location, positioning on the character block, and height of the cursor is programmable through registers RA, RB, RE, and RF. Figure 17- 10 illustrates how the cursor height is controlled by setting the horizontal scan line equal to the contents of RA and RB.

The underline register R14 controls the underline position in the character box. Set the value of R14 to be one less than the scan line counter for the desired row.

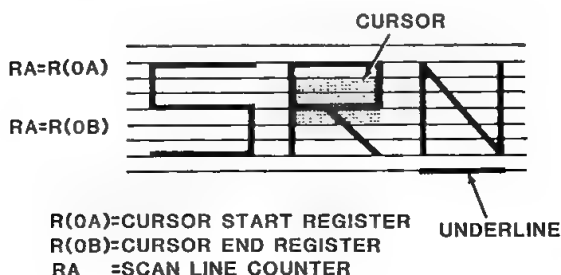


Figure 17-10. Cursor Control

The CRT controller provides the proper drive signals for a raster-scan display device. The 26 internal registers of the 82C434 can be software programmed to define display and control parameters. One of these registers, the address register, is used only as a pointer to the addresses of the other 25 registers. Table 17-62 summarizes the function of the CRT controller registers.

Table 17-62. 82C434 CRT Controller Register Summary

PORT ADDRESS	REGISTER NUMBER	REGISTER NAME
3?4	—	Address register (00-18)
3?5	R0	Horizontal total register
3?5	R1	Horizontal display enable end register
3?5	R2	Start horizontal blanking register
3?5	R3	End horizontal blanking register
3?5	R4	Start horizontal retrace register
3?5	R5	End horizontal retrace register
3?5	R6	Vertical total register
3?5	R7	Overflow register
3?5	R8	Preset row scan register
3?5	R9	Maximum scan line register
3?5	RA	Cursor start register
3?5	RB	Cursor end register
3?5	RC	High start address register
3?5	RD	Low start address register
3?5	RE	High cursor location register
3?5	RF	Low cursor location register
3?5	R10	Vertical retrace start register
3?5	R11	Vertical retrace end register
3?5	R10	High light pen register
3?5	R11	Low light pen register
3?5	R12	Vertical display enable end register
3?5	R13	Offset register
3?5	R14	Underline location register
3?5	R15	Start vertical blanking register
3?5	R16	End vertical blanking register
3?5	R17	Mode control register
3?5	R18	Line compare register
3?5	—	Mode register (see Table 17-66)

NOTE: The ? in the port address is B in monochrome modes and D in color modes.

31 kHz Video

Address Register — The address register is used to point to the other registers in the 82C434 and route input and output to the indicated register through port 3B5 in the monochrome modes or 3D5 in the color modes. The five least significant bits determine the register selected. Table 17-63 defines the address of each register.

Table 17-63. CRT Controller Addressing

BIT 4	BIT 3	BIT 2	BIT 1	BIT 0	VALUE	REGISTER
0	0	0	0	0	00	R0 — horizontal total register
0	0	0	0	1	01	R1 — horizontal display enable end register
0	0	0	1	0	02	R2 — start horizontal blanking register
0	0	0	1	1	03	R3 — end horizontal blanking register
0	0	1	0	0	04	R4 — start horizontal retrace register
0	0	1	0	1	05	R5 — end horizontal retrace register
0	0	1	1	0	06	R6 — vertical total register
0	0	1	1	1	07	R7 — overflow register
0	1	0	0	0	08	R8 — preset row scan register
0	1	0	0	1	09	R9 — maximum scan line register
0	1	0	1	0	0A	RA — start cursor register
0	1	0	1	1	0B	RB — end cursor register
0	1	1	0	0	0C	RC — high start address register
0	1	1	0	1	0D	RD — low start address register

Table 17-63 (continued). CRT Controller Addressing

BIT 4	BIT 3	BIT 2	BIT 1	BIT 0	VALUE	REGISTER
0	1	1	1	0	0E	RE — high cursor location register
0	1	1	1	1	0F	RF — low cursor location register
1	0	0	0	0	10	R10 — start vertical retrace/high light pen register
1	0	0	0	1	11	R11 — End vertical retrace/low light pen register
1	0	0	1	0	12	R12 — vertical display end register
1	0	0	1	1	13	R13 — offset register
1	0	1	0	0	14	R14 — underline location register
1	0	1	0	1	15	R15 — start vertical blank register
1	0	1	1	0	16	R16 — end vertical blank register
1	0	1	1	1	17	R17 — mode control register
1	1	0	0	0	18	R18 — line compare register

The partial program in Listing 17-4 illustrates a method that you may use to place a value (previously loaded into the CPU register BL) into the horizontal total register of the 82C434 CRT controller. With the exception of the mode register, loading data into other registers follows a similar pattern.

Listing 17-4. 82C434 Programming Example

```

MOV DX, 3D4H    ;select the register
MOV AL, 0       ;port address of the address register
OUT DX, AL      ;select the register for the data
                ;send AL to port defined by DX
                ;send the data
MOV DX, 3D5H    ;port address for internal register
MOV AL, BL      ;place value previously loaded in BL into AL
OUT DX, AL      ;send value to register

```

Horizontal Total Register (R0) — The value in the horizontal total register defines the total number of characters in a horizontal scan line, including retrace time. The value in the register is equal to the total number of characters less 2. All horizontal and vertical time periods are based upon the contents of this register. R0, when combined with the value in the retrace timing registers (R4 and R5), determine the period of the retrace output signal. The value of the character counter is compared with the value in this register to provide horizontal timing.

Horizontal Display Enable End Register (R1) — The value in the horizontal display register defines the length of the horizontal display enable signal and determines the number of characters to be displayed on one horizontal line. The actual number of characters displayed will be one less than the value placed in this register.

Start Horizontal Blanking Register (R2) — The value in the start horizontal blanking signal will become active. The value is defined in terms of when the number (of horizontal character clocks) in this register is equal to the horizontal character count. At that point, the blanking signal will become active.

The underline scan line decode output is multiplexed on the cursor output during the blanking period. The underline signal remains valid for one character count beyond the end of the blanking signal.

End Horizontal Blanking Register (R3) — The value in this register has two functions: (1) Provide an end horizontal blanking value (bits 0 through 4) and (2) provide skew control (bits 5 and 6). Bit 7 is not used.

The end horizontal blanking value is compared with the five least significant bits of the horizontal scan line register. When they agree, the horizontal blanking signal goes inactive. Because this value is limited to five bits, the maximum limit of the blanking signal is 31 character clocks. The value in this register is the value in the start blanking register plus the width of the blanking signal.

The display enable skew control value allows for character skew. Prior to displaying data on the screen, the CRT controller accesses the video buffer to obtain a character to be displayed, accesses the attribute code, accesses the character generator, and finally, reads the pixel panning register in the 82C433 attributes controller. Each access requires the display enabled signal to be skewed (delayed) by one character clock to allow synchronization with the horizontal and vertical retrace signals. The binary value in bits 5 and 6 of this register provide between 0 and 3 character clocks of skew or delay.

Start Horizontal Retrace Register (R4) — The value in this register is the character count at which time the horizontal retrace signal becomes active. It is used to center the display horizontally.

End Horizontal Retrace Register (R5) — The contents of this register perform three functions: (1) End horizontal retrace (bits 0 through 4), (2) Provide horizontal retrace delay (bits 5 and 6), and (3) Determine whether the start odd/even memory address is odd or even (bit 7).

During the end horizontal retrace function, the horizontal retrace signal goes inactive when the value in the five least significant bits of the horizontal scan line counter becomes equal to the count in this register. The value in the register is the value in the start register plus the width of the retrace signal in character clocks.

During the horizontal retrace delay function, some video modes require a horizontal retrace signal that takes the entire blanking period. The falling edge of this signal also provides some internal timing. To make sure that associated signals are properly latched, the retrace signal is started before the end of the display enable signal. To properly center the display on the screen, the skew (the horizontal retrace delay) needed is controlled by bits 5 and 6 of this register.

During the start odd/even memory address function where horizontal pixel panning is required, it may be necessary to start the video memory access at an odd address. Bit 7, if clear, selects an even address; if set, an odd address is selected. In most cases this bit should be set to 0.

31 kHz Video

Vertical Total Register (R6) — The vertical total register contains the least significant 8 bits of a nine-bit register. The ninth bit is contained in bit 0 of R7. The value of the vertical total register defines the number of horizontal scan lines in a single video frame, including the vertical retrace.

Overflow Register (R7) — This register contains the ninth bit of several other control registers and is used in conjunction with those registers. The registers are defined in Table 17-64.

Table 17-64. Overflow Register Contents

BITS	REGISTER	DESCRIPTION
0	R6	The ninth bit of the vertical total register.
1	R11	The ninth bit of the vertical display enable end.
2	R10	The ninth bit of the vertical retrace start register.
3	R15	The ninth bit of the vertical blanking register.
4	R18	The ninth bit of the line compare register.
5	RA	The ninth bit of the cursor location register.
6 and 7		Not used.

Preset Row Scan Register (R8) — Each horizontal retrace increments the horizontal row scan counter, which is cleared each time the counter equals the value in R9, the maximum scan line register. The value in the first five bits (bits 0 through 4) of this register (R8) specifies the starting row scan count after a vertical retrace.

In Hercules and color graphics compatible modes, this register can be used for soft scrolling by setting the register value between 0 and 3. For example, if you set the start scan row value at 1 instead of 0, the next frame will begin with address 1. This provides the effect of shifting the screen image vertically by 1 scan row. Bits 5 through 7 are not used.

Maximum Scan Line Register (R9) — The value in bits 0 through 4 of this register specifies the number of scan lines in each row of characters, minus one. Bits 5 through 7 are not used.

Cursor Start Register (RA) — The value in bits 0 through 4 of this register specifies the scan line in the character row (minus one) where a cursor is to begin. Bits 5 through 7 are not used.

Cursor End Register (RB) — The register performs two functions: (1) it defines the ending scan line of the cursor and (2) it provides cursor character skew.

During the cursor scan line end function, the value in bits 0 through 5 defines the scan line where the cursor ends. The start and end registers (RA and RB) allow a cursor of up to 32 scan lines in height to be started on any scan line of a character block. Usually the cursor will be no more than 14 scan lines in height, which is the height of the character block in mode 7 (monochrome card mode).

During the cursor skew function, the value of bits 6 and 7 determine how many characters to the right, the cursor will be skewed (offset). Zero, one, two, or three characters may be specified. Bit 7 of this register is not used.

Start Address Registers (RC and RD) — The value in the 16 bits that make up the RC and RD registers specifies the starting (first) address in video memory of the displayed page. Register RC contains the 8 most-significant (high-order) bits of the address while register RD contains the 8 least-significant (low-order) bit of the address.

Cursor Location Registers (RE and RF) — The value in the 16 bits that make up the RE and RF registers specifies the character address in video memory of the cursor. Register RE contains the 8 most-significant (high-order) bits of the address while register RF contains the 8 least-significant (low-order) bits of the address. By specifying the address anywhere in the 64K of video memory, paging and scrolling the display through memory will not lose the original cursor location.

Vertical Retrace Start Register (R10) — The vertical retrace start write-only register contains the least-significant 8 bits of a nine-bit register. The ninth bit is contained in bit 2 or R7. The value in these nine bits defines the horizontal scan line where the vertical retrace will start.

Vertical Retrace End Register (R11) — This write-only register serves several functions: (1) vertical retrace end, (2) clear vertical interrupt, (3) set vertical interrupt, and (4) test.

During the vertical retrace end function, the value in bits 0 through 3 specifies the horizontal scan line where the vertical retrace will end by comparing the value in this register with the four least-significant bits of the horizontal scan line counter.

During the clear vertical interrupt function, clearing bit 4 (setting it to 0) will clear the vertical interrupt generated on the CRTINT output of the CRT controller.

During the set vertical interrupt function, clearing bit 5 (setting it to 0) will enable the vertical interrupt signal on the CRTINT output of the CRT controller.

During the test function, bit 6 is used for testing the CRT controller during manufacturing, and must be cleared for normal operation (set to 0). Bit 7 of this register is not used.

Light Pen Registers (R10 and R11) — The value returned in the 16 bits that make up the R10 and R11 read-only registers is the video memory address after an attached light pen is triggered. Register RC contains the 8 most-significant (high-order) bits of the address while register RD contains the 8 least-significant (low-order) bits of the address.

Vertical Display Enable End Register (R12) — The vertical display enable end register contains the least-significant 8 bits of the nine-bit register. The ninth bit is contained in bit 1 of R7. The value in these nine bits defines the horizontal scan line where the display on the screen ends.

Offset Register (R13) — The value in the offset register defines the width of a logical line on the screen. The starting address of the next row is determined by the current row's starting address and the value in this register.

The video memory address counter can be incremented in 8-bit bytes or 16-bit words as determined by bit 6 of the mode control register, R17.

When the video memory address counter is counting in bytes, the next row's starting address is equal to the current row's starting address plus two times the value in R13. The least significant bit of the current row's address is forced to a zero at the input of the adder that sums the two values together.

When the video memory address counter is counting in words, the next row's starting address is equal to the current row's starting address plus four times the value in R13.

Underline Location Register (R14) — The value in bits 0 through 4 define the scan line of the character where the underline will appear. This value is one less than the actual scan line. Bits 5 through 7 are not used.

Start Vertical Blanking Register (R15) — The start vertical blanking register contains the least-significant 8 bits of a nine-bit register. The ninth bit is contained in bit 3 of R7. The value in these nine bits defines the horizontal scan line where the vertical blanking signal starts.

End Vertical Blanking Register (R16) — The value in bit 0 through bit 4 of the end vertical blanking register is compared against the value of the five least-significant bits in the horizontal scan line counter. When these two values match, vertical blanking will be terminated. The maximum width of the vertical blanking signal is limited to 31 horizontal scan lines from the start of the vertical display. Bits 5 through 7 are not used.

31 kHz Video

Mode Control Register (R17) — The mode control register provides control over a number of functions these are described in Table 17-65.

Table 17-65. Mode Control Register Functions

BIT	DESCRIPTION
0	Compatibility mode support. This bit allows compatibility with the 6845 CRT controller. When this bit is clear, scan row address bit 0 is substituted for memory address bit 13 during active display times and provides 6845 compatibility. When this bit is set, no such substitution takes place.
1	Select scan row counter. This bit allows compatibility with 400-line graphics cards, such as the Hercules graphics card. When this bit is clear, the MA14 output signal reflects the value in bit 1 of the scan row counter. When this bit is set, the MA14 output signal reflects the value in bit 14 of the scan row counter.
2	Horizontal retrace select. This bit controls the horizontal resolution of the CRT controller. When this bit is clear, the vertical counter follows the horizontal retrace clock which provides a maximum horizontal resolution of 512 scan lines. When this bit is set, the vertical counter is advanced every other horizontal retrace clock, providing a maximum horizontal resolution of 1,024 scan lines.
3	Count by two. This bit determines whether the value in the offset register (R13) is a byte or word value. When this bit is clear, the memory address counter is clocked by the character clock input and the value in R13 is in bytes. When this bit is set, the memory address counter is advanced every other character clock and the value in R13 is in words.
4	Output control. This bit controls the state of these CRT controller output signals: VRTC, SYNC, BLANK, CURSOR, DE, MA14 and MA15. When this bit is clear, the output lines are enabled. When this bit is set, the output lines are placed into a high impedance state.

Table 17-65 (continued). Mode Control Register Functions

BIT	DESCRIPTION
5	Address Wrap. This bit determines whether MA13 or MA15 appears on the MA0 output pin when bit 6 is clear (has selected the word mode). When both bit 5 and bit 6 are clear, MA13 will be selected; this indicates that video memory is limited to 64K. When bit 6 is clear and bit 5 is set, MA15 will be selected, indicating that video memory is greater than 64K. Since video memory is 256K on this card, MA15 should always be selected whenever bit 6 is clear. When bit 6 is set, the byte mode has been selected and the MA0 counter output has been selected and appears on the MA0 output pin.
6	Word or byte mode. When this bit is set, the byte mode is selected and the memory address counter bits are in their normal position. When this bit is clear, the word mode is selected and the memory address counter bits are shifted one position to the right, causing the most significant bit of the memory address counter to appear as the least significant bit at the memory address outputs.
7	Hardware reset. When this bit is clear, the vertical and horizontal retrace registers are cleared. When this bit is set, the vertical and horizontal retrace registers are enabled.

Line Compare Register (R18) — The line compare register contains the least-significant 8 bits of a nine-bit register. The ninth bit is contained in bit 4 of R7.

The nine-bit line compare register is used in split screen applications. The scrolling operation utilizes the start address registers RC and RD. The split screen capability will allow scrolling through some areas of the screen while the remaining screen area remains immune to it.

When the horizontal scan line counter is equal to the value in this register, the memory address counter is reset to 0. The counter then sequentially addresses the display buffer, starting with address 0. Each subsequent row's starting address is determined by summing the start-of-line latch with the value in the offset register. This allows a given area of the screen to remain immune from scrolling.

31 kHz Video

Mode Register — The mode register in the CRT controller is written to by activating the MODEIOW input to the CRT controller. The mode control register uses the three most significant bits (5 through 7) to perform the functions in Table 17-66. Bits 0 through 4 are not used.

Table 17-66. Mode Register Functions

BIT	DESCRIPTION
5	This bit is the least significant bit of the video memory address when the inputs SAM and MEMOPT are low and CHAIN is high.
6	This bit selects the polarity of the horizontal sync signal. If this bit is clear, the signal is positive. If this bit is set, the signal is negative.
7	This bit selects the polarity of the vertical sync signal. If this bit is clear, the signal is positive. If this bit is set, the signal is negative.

82C434 Pinout

The 82C434 CRT controller is an 84-pin, large-scale integrated circuit. The pin definitions are described in Table 17-67 and illustrated in Figure 17-11.

Table 17-67. 82C434 CRT Controller Pin Functions

PIN NUMBER	SIGNAL NAME	DEFINITION
1	Vss	Ground.
2	AAV14	Video address bus A line 14. Generated by the 82C432 sequencer.
3	BAV14	Video address bus B line 14. Generated by the 82C432 sequencer.
4	BA0	Video address bus B line 0. Used to address memory planes 2 and 3.

Table 17-67 (continued). 82C434 CRT Controller Pin Functions

PIN NUMBER	SIGNAL NAME	DEFINITION
5	A1	CPU address bit 1. Used to generate addresses for the video memory.
6	AA1	Video address bus A line 1. Used to address memory planes 0 and 1.
7	A9	CPU address bit 9. Used to generate addresses for the video memory.
8	BA1	Video address bus B line 1. Used to address memory planes 2 and 3.
9	M0D4	Memory plane 0, bit 4. Video memory bit, which is part of the character data bus.
10	AA2	Video address bus A line 2. Used to address memory planes 0 and 1.
11	A2	CPU address bit 2. Used to generate addresses for the video memory.
12	A10	CPU address bit 10. Used to generate addresses for the video memory.
13	BA2	Video address bus line 2. Used to address memory planes 2 and 3.
14	M0D5	Memory plane 0, bit 5. Video memory bit, which is part of the character data bus.
15	AA3	Video address bus A line 3. Used to address memory planes 0 and 1.
16	A3	CPU address bit 3. Used to generate addresses for the video memory.
17	A11	CPU address bit 11. Used to generate addresses for the video memory.

31 kHz Video

Table 17-67 (continued). 82C434 CRT Controller Pin Functions

PIN NUMBER	SIGNAL NAME	DEFINITION
18	BA3	Video address bus B line 3. Used to address memory planes 2 and 3.
19	M0D6	Memory plane 0, bit 6. Video memory bit, which is part of the character data bus.
20	$\overline{\text{LG}}$	Line graphics. This active low output is used by the 82C433.
21	Vdd	+ 5 VDC supply.
22	Vss	Ground.
23	A4	Video address bus A line 4. Used to address memory planes 0 and 1.
24	A4	CPU address bit 4. Used to generate addresses for the video memory.
25	A12	CPU address bit 12. Used to generate addresses for the video memory.
26	BA4	Video address bus B line 4. Used to address memory planes 2 and 3.
27	M0D7	Memory plane 0, bit 7. Video memory bit, which is part of the character data bus.
28	AA5	Video address bus A line 5. Used to address memory planes 0 and 1.
29	A5	CPU address bit 5. Used to generate addresses for the video memory.
30	A13	CPU address bit 13. Used to generate addresses for the video memory.
31	BA5	Video address bus B line 5. Used to address memory planes 2 and 3.

Table 17-67 (continued). 82C434 CRT Controller Pin Functions

PIN NUMBER	SIGNAL NAME	DEFINITION
32	MOD0	Memory plane 0, bit 0. Video memory bit, which is part of the character data bus.
33	AA6	Video address bus A line 6. Used to address memory planes 0 and 1.
34	A6	CPU address bit 6. Used to generate addresses for the video memory.
35	A8	CPU address bit 8. Used to generate addresses for the video memory.
36	BA6	Video address bus B line 6. Used to address memory planes 2 and 3.
37	MOD3	Memory plane 0, bit 3. Video memory bit, which is part of the character data bus.
38	MOD1	Memory plane 0, bit 1. Video memory bit, which is part of the character data bus.
39	AA7	Video address bus A line 7. Used to address memory planes 0 and 1.
40	A7	CPU address bit 7. Used to generate addresses for the video memory.
41	AAV15	Video address bus A line 15. Generated by the 82C432 sequencer.
42	Vdd	+ 5 VDC supply.
43	Vss	Ground.
44	BA7	Video address bus B line 7. Used to address memory planes 2 and 3.
45	MOD2	Memory plane 0, bit 2. Video memory bit, which is part of the character data bus.

31 kHz Video

Table 17-67 (continued). 82C434 CRT Controller Pin Functions

PIN NUMBER	SIGNAL NAME	DEFINITION
46	BAV15	Video address bus B line 15. Generated by the 82C432 sequencer.
47	MA15	Memory address 15. This output is used by the 82C432 to generate the high order address bits AAV15 and BAV15. This output can be placed into a high-impedance mode by mode control register R17.
48	MA14	Memory address 14. This output is used by the 82C432 to generate the high order address bits AAV14 and BAV14. This output can be placed into a high-impedance mode by mode control register R17.
49	<u>GRAPHICS</u>	Graphics. This active low input from the 82C431 is used to change RAM BA addresses from graphics to text mode.
50	CHAIN	Chain. This input from the 82C431 is used with the SAM and MEMOPT inputs to select the LSB of the CPU address.
51	MUX	Multiplex. This input from 82C432 is used to multiplex the RAS and CAS address control signals.
52	RESET	Reset. This signal initializes the CRTC as follows: (1) horizontal and vertical polarity control are cleared to 0, (2) mode register bits 4 and 7 are cleared to 0, (3) all counters in the CRT controller are reset, (4) address register bits 3 and 4 are cleared to 0, and (5) PGSEL is cleared to 0.
53	HIN	Horizontal sync. This output is active high if the horizontal polarity bit HPOL is high. HPOL is programmed through the MODEIOW register. This signal is identical to the HPOL bit of the miscellaneous output register in the 82C431.

Table 17-67 (continued). 82C434 CRT Controller Pin Functions

PIN NUMBER	SIGNAL NAME	DEFINITION
54	$\overline{\text{CRTL}}$	CRT latch. This a active low input from the 82C432 latches data bit 0 through data bit 7 of memory plane 0.
55	DE	Display enable. This output enables the display area on the screen. When DE and BLANK are inactive at the same time, the border is displayed on the monitor.
56	CURSOR	Cursor. This output signal serves two functions. First, it indicates a valid cursor position when DE is active. Second, it signifies the contents of the underline location register R14 are equal to the horizontal line counter.
57	BLANK	Blank. This output signal is used to blank the screen during retrace periods.
58	$\text{CRT}/\overline{\text{CPU}}$	CRT / CPU. When this input signal is high, the CRT read address signal is enabled. When this input signal is low, the CPU read address or CPU write address signal is enabled.
59	VIN	Vertical sync. This output is active high if the horizontal polarity bit VPOL is high. VPOL is programmed through the MODEIOW register. This signal is identical to the VPOL bit of the miscellaneous output register in the 82C431.
60	SYNCS	Sync. This output is used by the 82C432 to control the AS/L signal.
61	VRTC	Vertical retrace. This output is used by the 82C433 to generate blink clock for the blinking portions of the display.
62	$\overline{\text{CLOCK}}$	Clock. This signal is CCLK from the 82C432 and is used to synchronize all functions for the 82C434 except the bus interface. The active transition is high to low.

Table 17-67 (continued). 82C434 CRT Controller Pin Functions

PIN NUMBER	SIGNAL NAME	DEFINITION
63	Vdd	+ 5 VDC supply.
64	Vss	Ground.
65	CRTINT	CRT interrupt. This is the frame interrupt output, which is enabled by bit 5 of the vertical retrace end register. It can be cleared by programming bit 4 of the vertical retrace end register to 0.
66	$\overline{\text{MODEIOW}}$	Mode input/output write. This active low input is used to program the MODEIOW register bits 5, 6, and 7. A chip reset clears the register.
67	$\overline{\text{IOW}}$	Input/output write. This active low input from the CPU is used to write to the 82C434 registers.
68	$\overline{\text{IOR}}$	Input/output read. This active low input from the CPU is used to read the 82C434 registers.
69	LPS	Light pen strobe. This input from the light pen is used to latch the current refresh address in the light pen register. The address is latched on a low-to-high transition of this input.
70	A0	CPU address bit 0. Used to generate addresses for the video memory.
71	SAM	This input is used with MEMOPT and CHAIN to select the LSB of the CPU address.
72	MEMOPT	Memory option. This input from the video memory is used with SAM and CHAIN to select the LSB of the CPU address. When this signal is low, video memory contains 64K.

31 kHz Video

31 KHz Interface

The 31 kHz video card is a high performance card capable of emulating standard CGA, MDA, Hercules, and EGA video formats. It also supplies a more advanced degree of resolution (640×480) than standard EGA designs. The card is supplied with two external connectors: a 9-pin connector for standard (native) video formats and a 15-pin connector that provides RGB analog signals.

In the native modes (CGA, MDA, Hercules, and EGA), various resolutions and capabilities are possible. The previous sections in the chapter describe the registers that support these functions. In the 31 kHz mode, up to 480 horizontal lines can be generated and 16 colors (out of a palette of 256,000 possible colors) can be displayed. The following sections describe the programmable registers of the 31 kHz extended chip set and its associated color palette chip.

31 kHz Extended Chip Set

The extended chip set consists of the 82C431 graphics controller, the 82C432 sequencer, the 82C433 attributes controller, the 82C434 CRT controller, and a custom gate array. The register descriptions for all of these devices (excluding the custom gate array) are provided in the "Enhanced Graphics" section of this chapter.

The custom gate array is a proprietary design and no information with regard to device programmability is available. The gate array decodes the appropriate signals to emulate the native mode video formats and to implement 31 kHz EGA (640×480).

Color Palette

The color palette is an integrated circuit that is capable of displaying 256 colors from a palette of 256,000 possible colors. The IMS-G171 color palette IC contains a color look-up table, three 6 bit DAC (digital-to-analog conversion) circuits, and processor interface circuits.

The device is used in the final stage of the video system. It contains sufficient memory to maintain the status of $256K \times 18$ bit data words. Each word is composed of three 6-bit data fields. The three DAC circuits convert each 6-bit field into R, G, and B video levels. These levels drive the appropriate type of video monitor to produce various colors (or shades of gray) on the monitor's display.

The following section provides a description of the internal registers of the color palette IC. Table 17-68 indicates which register can be accessed according to the status of the register-select (RS0 and RS1) signals. The registers can be read or written to in accordance with the states of the read enable and write enable signals.

Table 17-68. Color Palette Registers

RS0	RS1	REGISTER	PORT
0	0	Write pixel address register	3C8H
0	1	Color value register	3C9H
1	0	Pixel mask register	3C6H
1	1	Read pixel address register	3C7H

Write Pixel Address Register — This register defines a starting address for write operations to the color palette. During a processor write cycle, the value placed in this register defines an memory location that contains a color definition. The color definition can be changed by writing new values to the color value register. After a new color value is written, the write pixel address register increments to point to the next memory location in the color palette.

A color value write cycle requires that three bytes be placed in the color value register. The first byte defines red color information, the second byte defines green color information, and the third byte defines blue color information. The first six least-significant bits of each byte are strung together in the color value register to form an 18-bit data segment.

If a new value is written to the pixel address register when a write operation (through the color value register) to the color palette is taking place, the operation will be terminated and the color values in the palette will be set to an initial condition.

31 kHz Video

Color Value Register — The color value register is an 18 bit register that temporarily reflects values that define colors in the palette. New color values can be written to the palette through the color value register and the status of the current palette can be read through the register.

During a write cycle, three bytes that define red, green, and blue color information are assembled in the 18-bit color value register. Note that only the first six least-significant bits of each byte are used. After the 18-bit data segment is assembled it is written to a specific memory location (defined by the write pixel address register) in the color palette.

During a read cycle, the color information in the palette is transferred into the 18-bit color value register. Each 6-bit data field is read as three bytes with the two most-significant bytes set to zero.

Pixel Mask Register — Before the color palette is addressed, each bit in the pixel mask register is logically ANDed with each bit that makes up the pixel address. This function can cause the displayed colors to be altered while the color definitions in the color palette and in video memory are not altered. Manipulating the pixel address as a function of the pixel mask register allows such effects as rapid animation and flashing of objects.

Read Pixel Address Register — This register defines a starting address for read operations to the color palette. During a processor read cycle, the value placed in this register defines a memory location that contains a color definition. The color definition can be read through the color value register. After a color value is read, the read pixel address register increments to point to the next memory location in the color palette.

A color value read cycle fetches three bytes from the color palette. The first byte defines red color information, the second byte defines green color information, and the third byte defines blue color information. The first six least-significant bits are fetched from the 18-bit string in the color value register. The two most-significant bits of each byte are set to zero. Each 6-bit data field (representing R, G, and B color information) is converted to analog levels by the IMS-G171's internal DAC circuits.

If a new value is written to the read pixel address register when a read operation (through the color value register) to the color palette is taking place, the operation will be terminated and the color values in the palette will be set to an initial condition.



Chapter 18

Serial and Parallel Communications

This chapter describes the communications hardware. The Advanced Desktop Computer is supplied with one serial port and one parallel port. The circuits that control the serial port consist of an NS16450 asynchronous communications element and various buffers and drivers that act as receivers and transmitters. Data transceivers handle the data bus interface. The parallel port consists of three I/O ports. The parallel port circuits do not include a designated VLSI integrated circuit that controls port operations. Instead, discrete chips interface parallel data to 8-bit parallel devices. The chips are designated as specific drivers and receivers.

NOTE: The serial port may be configured as COM1 or COM2 by setting the appropriate address and interrupt jumpers. Since there is only one address jumper, the serial port and the parallel port must be set for COM1 and LPT1 or COM2 and LPT2. If the communications ports are set for COM1, the interrupt jumper should be set for IRQ4. If the communications ports are set for COM2, the interrupt jumper should be set for IRQ3. Refer to Chapter 4 for additional configuration information.

The following sections describe the hardware and programmable characteristics (where applicable) of the integrated circuits used for data communication.

Serial Port

In this computer, the serial port circuitry is on the I/O card, which is located in the third slot of the backplane board. The serial communication signals are transmitted through a 9-pin connector. Table 18-1 details which signals are carried by which pins.

Serial and Parallel Communications

Table 18-1. Serial Connector Pinout

PIN NUMBER	SIGNAL NAME	DESCRIPTION
1	CD	Carrier detect
2	SIN	Receive data
3	SOUT	Transmit data
4	DTR	Data terminal ready
5	GND	Signal ground
6	DSR	Data set ready
7	RTS	Request to send
8	CTS	Clear to send
9	RI	Ring indicate
Case	GND	Chassis ground

The serial port hardware is designed around the NS16450 VLSI controller chip. This device operates very much like the INS8250A serial port controller chip. It is bidirectional to permit I/O communication with serial peripherals. Parallel-to-serial (data bus to device) and serial-to-parallel (device to data bus) data conversion, parity, and handshaking are all handled by the NS16450.

NS16450 Asynchronous Communications Element

The asynchronous communications element controls the serial communications channel. A block diagram of this device is shown in Figure 18-1.

The NS16450 interfaces the CPU with the RS-232C serial input/output connector on the back of the computer. The controller receives information from the system data bus through an internal data buffer. The NS16450 can be programmed with this data or transmit it through the port. Addressing is handled by the chip select line and address lines 0 – 2. Control logic inside the controller determines if a register is being programmed or if there is data ready to transmit or receive.

Serial and Parallel Communications

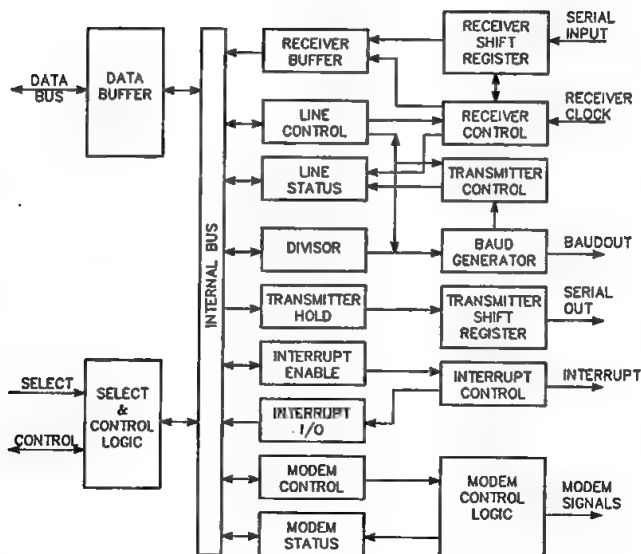


Figure 18-1. NS16450 Block Diagram

The receiver circuits accept serial data from pin 2 (serial in) of the connector and convert it into a parallel form before making it available for the system data bus. The transmitter circuits convert parallel data from the system data bus into serial form and then send it to pin 3 (serial out) of the connector.

The modem support circuits provide the control signals used for serial communications across standard telephone lines. The system interrupt controller signals the processor when the NS16450 requires service to continue transmitting or receiving data. In this computer, this line is only used to request service while receiving data.

The NS16450 provides a number of programmable options: manipulating interrupts, modem controls, word length, parity, number of stop bits, and baud rate. Most of these functions can be handled at the system level through the interrupts described in Chapter 9.

Serial and Parallel Communications

Handshaking

Two devices communicate asynchronously through handshaking. Two handshaking methods are used: software and hardware. This part of the manual is concerned only with hardware handshaking. For information on software handshaking, refer to Chapter 9.

The four input handshaking lines at the NS16450 are $\overline{\text{RLSD}}$, $\overline{\text{DSR}}$, $\overline{\text{CTS}}$ and $\overline{\text{RI}}$. The $\overline{\text{RLSD}}$ (receive line signal detect) line is connected to the carrier detect line on the RS-232C connector. It is asserted whenever an externally connected modem detects an incoming carrier. The $\overline{\text{DSR}}$ (data set ready) line is asserted when a modem or other device is ready to establish communications with the computer. The $\overline{\text{CTS}}$ (clear to send) line is asserted when the external device is ready to accept data. The $\overline{\text{RI}}$ (ring indicator) line is asserted when an attached modem detects a ring signal on the telephone line. All signals are inverted between the connector and NS16450 by an inverter/receiver.

The two output handshake lines are $\overline{\text{DTR}}$ and $\overline{\text{RTS}}$. The $\overline{\text{DTR}}$ (data terminal ready) line is asserted when the computer is ready to communicate with an external device. The $\overline{\text{RTS}}$ (request to send) line is asserted when the computer is ready to transmit data. These two lines are buffered by drivers between the NS16450 and the connectors.

NS16450 Programming

NOTE: The original 8250 design had a bug in it that required programming the 8250 while in the loopback mode. Since this bug has been corrected in the device used in this computer (NS16450), the practice of placing the device in the loopback mode is no longer necessary. However, to maintain software compatibility with older computers that use the earlier versions of the 8250, you may wish to continue this practice.

Serial and Parallel Communications

There are nine registers available to the programmer in the NS16450. Refer to Table 18-2 for the port addresses of these registers.

Table 18-2. NS16450 Register Ports

ADDRESS	REGISTER
3F8H	Receiver buffer register
3F8H	Transmitter holding register
3F8H	Divisor latch (least-significant byte)
3F9H	Divisor latch (most-significant byte)
3F9H	Interrupt enable register
3FAH	Interrupt identification register
3FBH	Line control register
3FCH	Modem control register
3FDH	Line status register
3FEH	Modem status register

NOTE: The port addresses referred to in Table 18-2 are normally associated with the COM1: serial communications port. If a second serial port is added to the system (COM2:) its addresses will begin at 2F8H.

The receiver buffer register and the transmitter holding register share the same port address. The receiver buffer register is a read-only register and the transmitter holding register is a write-only register. These registers contain the data received and the data to be transmitted. The first data bit to be transmitted or received is the least-significant bit.

The divisor latches establish the baud rate based on the input frequency of the oscillator clock to the NS16450. This computer uses an oscillator with a frequency of 1.8432 MHz. You can program the latches with any number to produce a value that is sixteen times the actual baud rate. The values that produce standard baud rates, along with the error percentage, are provided in Table 18-3.

NOTE: To write to the divisor latches, you must first set bit 7 in the line control register. Otherwise, the receiver buffer, the transmitter holding register, or the interrupt enable register will be written to.

Serial and Parallel Communications

Table 18-3. Baud Rate and Divisor Latch Values

DESIRED BAUD RATE	DIVISOR VALUE	ERROR PERCENTAGE
50	2304	0
75	1536	0
110	1047	0.026
134.5	857	0.058
150	768	0
300	384	0
600	192	0
1200	96	0
1800	64	0
2000	58	0.69
2400	48	0
3600	32	0
4800	24	0
7200	16	0
9600	12	0
19200	6	0
38400	3	0
56000	2	2.86

NOTE: Although any baud rate can be programmed into the NS16450, PC-compatible hardware and software supports only the following baud rates: 110, 150, 300, 600, 1200, 2400, 4800, 9600, and 19,200.

The two interrupt registers work together but have different port addresses. The interrupt enable register uses four bits to enable the interrupt capabilities of the NS16450. Four interrupts are recognized: received data available interrupt, transmitter holding register empty interrupt, receiver line status interrupt, and modem status interrupt. These interrupts are identified by the first three bits of the interrupter identification register. Their priority level, source, and reset control are described in Table 18-4.

Serial and Parallel Communications

Table 18-4. NS16450 Interrupts

INTERRUPT REGISTER VALUE	ID PRIORITY LEVEL	SOURCE	RESET CONTROL
0	4	Clear to send, data set ready, ring indicator, or received line signal detect.	Read modem status register.
1	—	None.	None.
2	3	Empty interrupt ID register or transmitter holding register.	Read interrupt ID register, or transmitter holding register, whichever caused the interrupt.
3	—	None.	None.
4	2	Received character available in the receiver buffer register.	Read the receiver buffer register.
5	—	None.	None.
6	1	Overflow, parity, or framing error, or break interrupt.	Read the line status register.
7	—	None.	None.

The line control register establishes many of the parameters for serial communications. Each bit is described in Table 18-5.

Serial and Parallel Communications

Table 18-5. Line Control Register

BIT	DESCRIPTION
0-1	Word length: 0 = 5 bits, 1 = 6 bits, 2 = 7 bits, and 3 = 8 bits. Bit 0 is the least-significant bit.
2	Number of stop bits dependent upon the word length. Word length = 6, 7, or 8: clear (0) = 1 stop bit, set (1) = 2 stop bits. Word length = 5: clear (0) = 1 stop bit, set (1) = 1.5 stop bits.
3	Parity enabled or not: clear (0) = parity disabled, set (1) = parity enabled.
4	Even or odd parity: clear (0) = odd parity, set (1) = even parity.
5	Stick parity. When parity (bit 3) and stick parity are both enabled, parity will be opposite that indicated by bit 4 (odd/even parity). Clear (0) = stick parity disabled, set (1) = stick parity enabled.
6	Set break control. When this bit is set (1), the serial output goes low (space) and remains there until this bit is cleared (0). This feature allows the computer to alert a remote station, or vice versa, in a data communications network.
7	Allows divisor latches to be accessed: clear (0) = latch access disabled, set (1) = the latches can be read from or written to.

The modem control register handles several parameters and some of the handshaking protocol for the NS16450. Each bit is described in Table 18-6.

Table 18-6. Modem Control Register

BIT	DESCRIPTION
0	This bit controls the data terminal ready output: clear (0) = $\overline{\text{DTR}}$ output is high, set (1) = $\overline{\text{DTR}}$ output is low. (See the note at the bottom of this table.)

Serial and Parallel Communications

Table 18-6 (continued). Modem Control Register

BIT	DESCRIPTION
1	This bit controls the request to send output: clear (0) = \overline{RTS} output is high, set (1) = \overline{RTS} output is low. (See the note at the bottom of this table.)
2	This bit controls the output 1 signal: clear (0) = $\overline{OUT1}$ output is high, set (1) = $\overline{OUT1}$ output is low.
3	This bit controls the output 2 signal in same manner that the output 1 signal is controlled. However, $\overline{OUT2}$ is pulled high in this computer and not used.
4	This bit provides a local loopback condition to test the NS16450. It is used by the disk-based diagnostics to check the data transmit and receive circuits of the NS16450.
5-7	Not used, permanently clear (0).

NOTE: The outputs described in this table are at the chip and do not necessarily reflect output signals at the connector.

The line status register provides a status report concerning data transfer. Each bit is described in Table 18-7.

Table 18-7. Line Status Register Report

BIT	DESCRIPTION
0	Indicates that a complete character has been received and transferred into the receiver buffer register: set (1) = character received and ready; clear (0) = receiver buffer register is empty (either no character has been received or the receiver buffer register has been read).
1	Indicates that the receiver buffer register was not read before the next character was received: set (1) = the next character has been received before the receiver buffer register was read; clear (0) = no overrun has taken place.
2	Indicates that a parity error has been detected: set (1) = parity error detected; clear (0) = no parity error.

Serial and Parallel Communications

Table 18-7 (continued). Line Status Register Report

BIT	DESCRIPTION
3	Indicates that a framing error has taken place, no valid stop bit was received: set (1) = framing error detected; clear (0) = no framing error.
4	Indicates that a break interrupt has been detected. A break interrupt occurs when the received data input is held at a logical 0 state longer than a full word time (start plus data plus parity plus stop bits).
5	Indicates that the transmitter holding register is empty: set (1) = transmitter holding register empty; clear (0) = transmitter holding register receiving or has a character. This bit will produce an interrupt to the CPU if the transmitter holding register empty interrupt is enabled.
6	Indicates when the transmitter shift register is idle: set (1) = shift register is idle; clear (0) = shift register has received a character or is active (shifting data out).
7	Not used, permanently clear (0).

NOTE: Bits 1 – 4 produce a receiver line status interrupt when one of the conditions is detected.

The modem status register provides a status report concerning the handshaking and control lines. Each bit is described in Table 18-8.

Table 18-8. Modem Status Register Report

BIT	DESCRIPTION
0	Indicates that the $\overline{\text{CTS}}$ signal has changed state since the last time this register was read: set (1) = changed state, clear (0) = no change.
1	Indicates that the $\overline{\text{DSR}}$ signal has changed state since the last time this register was read: set (1) = changed state, clear (0) = no change.

Serial and Parallel Communications

Table 18-8 (continued). Modem Status Register Report

BIT	DESCRIPTION
2	Indicates that the \overline{RI} signal has changed from a mark (set or logical 1) condition to a space (clear or logical 0) condition: set (1) = changed state, clear (0) = no change.
3	Indicates that the \overline{RLSD} signal has changed state since the last time this register was read: set (1) = changed state, clear (0) = no change.
4	Reflects the complement of the \overline{CTS} input: set (1) = \overline{CTS} clear (0), clear (0) = \overline{CTS} set (1).
5	Reflects the complement of the \overline{DSRL} input: set (1) = \overline{DSR} clear (0), clear (0) = \overline{DSR} set (1).
6	Reflects the complement of the \overline{RI} input: set (1) = \overline{RI} clear (0), clear (0) = \overline{RI} set (1).
7	Reflects the complement of the \overline{RLSD} input: set (1) = \overline{RLSD} clear (0), clear (0) = \overline{RLSD} set (1).

NOTE: Bits 0 – 3 produce a modem status interrupt when one of the conditions is detected and the corresponding bit is set.

Pinout

The pin functions of the NS16450 are described in Table 18-9 and the pinout is illustrated in Figure 18-2.

Table 18-9. NS16450 Pin Functions

PIN NUMBER	SIGNAL NAME	DESCRIPTION
1 – 8	D0 – D7	Data line 0 – data line 7. These three-state, bidirectional lines are connected to the computer's data bus.

Serial and Parallel Communications

Table 18-9 (continued). NS16450 Pin Functions

PIN NUMBER	SIGNAL NAME	DESCRIPTION
9	RCLK	Receive clock. This input is tied to pin 15, the baud out signal. It provides the timing for the baud rate of the receiving circuits inside the NS16450. By tying this line to the baud out signal, the transmitting and receiving baud rates remain the same.
10	SIN	Serial input. This input receives the input signal from the connector. It is buffered by an external inverter/buffer.
11	SOUT	Serial output. This output line transmits the serial data to the connector through a pair of external buffer/drivers.
12 – 13	CS0 – CS1	Chip select lines 0 – 1. These two lines are tied high. Selection of the serial port is handled by pin 14.
14	$\overline{\text{CS2}}$	Chip select. This input signal comes from the decoder, described in Chapter 14. It is used to select the serial port.
15	$\overline{\text{BAUDOUT}}$	Baud out. This signal is a clock signal that runs at sixteen times the transmitted baud rate. This output is tied to pin 9, the receive clock, to make the transmit and receive baud rates identical.
16	XTAL 1	External clock in. This is one side of the 1.8432 MHz crystal oscillator. It is used as the frequency base for the various baud rates that can be programmed for the NS16450.
17	XTAL 2	External clock out. This is the other side of the 1.8432 MHz crystal oscillator. The output is used to stabilize the oscillator.
18	$\overline{\text{DOSTR}}$	Data out strobe. This line is connected to the system $\overline{\text{IOW}}$ line, which is used to write data to the selected NS16450 register.

Serial and Parallel Communications

Table 18-9 (continued). NS16450 Pin Functions

PIN NUMBER	SIGNAL NAME	DESCRIPTION
19	DOSTR	Data out strobe. Tied low and not used.
20	GND	Ground.
21	$\overline{\text{DISTR}}$	Data in strobe. This line is connected to the system $\overline{\text{IOR}}$ line, which is used to read the contents of the selected NS16450 register.
22	DISTR	Data in strobe. Tied low and not used.
23	DDIS	Driver disable. Not connected.
24	CSOUT	Chip select out. Not connected.
25	ADS	Address strobe. This input provides latching for the address and chip select lines. Since these inputs are stable in this computer, this line is tied low.
26 – 28	A2 – A0	Address line 0 – address line 2. These lines provide register addressing information for the NS16450.
29	—	Not used.
30	INTRPT	Interrupt. This output is ANDed with the $\overline{\text{IOR}}$ line to produce IRQ4 for the system. $\overline{\text{OUT2}}$ activates the interrupt.
31	$\overline{\text{OUT2}}$	Output 2. This output is programmed by the system to control IRQ4 selection.
32	$\overline{\text{RTS}}$	Request to send. This output indicates the NS16450 is ready to send information. It is a programmed handshaking signal.
33	$\overline{\text{DTR}}$	Data terminal ready. This output indicates the NS16450 is read to communicate. It is a programmed handshaking signal.
34	$\overline{\text{OUT1}}$	Output 1. This output is pulled high and not used in this computer.

Serial and Parallel Communications

Table 18-9 (continued). NS16450 Pin Functions

PIN NUMBER	SIGNAL NAME	DESCRIPTION
35	MR	Master reset. This input line is tied to RESET and clears and resets the NS16450's registers.
36	$\overline{\text{CTS}}$	Clear to send. This input indicates that the remote device is ready to transmit. It is a programmed handshaking signal.
37	$\overline{\text{DSR}}$	Data set ready. This input indicates the status of the data set. It is a programmed handshaking signal.
38	$\overline{\text{RLSD}}$	Received line signal detect. This input is tied to the connector's carrier detect input pin and indicates that an acceptable-quality signal carrier has been detected. It is a programmed handshaking signal.
39	$\overline{\text{RI}}$	Ring indicator. This input is tied to the serial connector's ring detect pin and indicates that a ringing signal has been detected. It is a programmed handshaking signal.
40	Vcc	+ 5 VDC power supply.

Serial and Parallel Communications

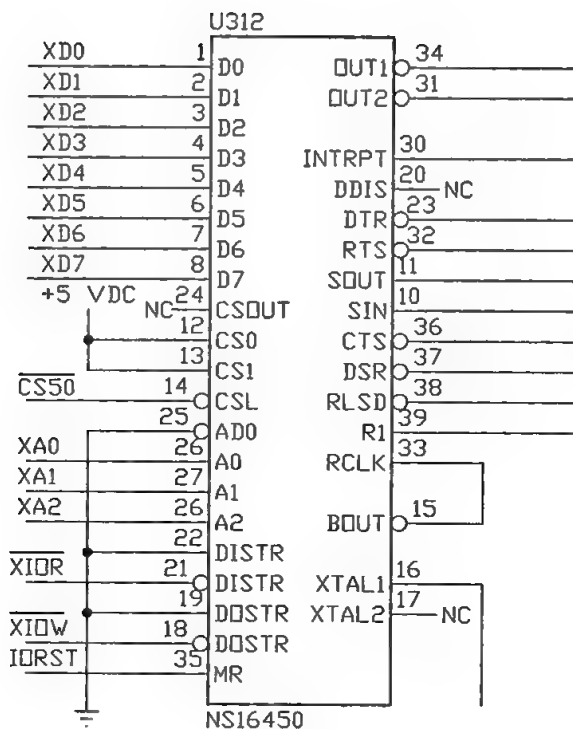


Figure 18-2. NS16450 Pinout

Parallel Port

The parallel port consists of three individual I/O ports (A, B, and C). These devices interface system data to peripheral 8-bit parallel equipment. I/O port A is a read/write data port that transmits data to the peripheral during a write operation. During a read operation, port A is addressed at 378-37F (LPT1) or 278-27F (LPT2). The same data previously transmitted is read. In other words, port A will not read data from the equipment. This procedure is done to make a check of the transmitted data. If transmitted data does not agree with received data, an error condition exists.

I/O port B is a read-only port addressed at 379H (LPT1) or 279H (LPT2). It allows polling of the following signals:

- Error status (parallel data bit 3)
- Select acknowledge (parallel data bit 4)
- Out of paper (parallel data bit 5)
- Data acknowledge (parallel data bit 6)
- Device busy (parallel data bit 7).

I/O port C is a read/write port that controls transfer of information to the peripheral equipment. It is addressed at 37AH (LPT1) or 27AH (LPT2). I/O port C is connected to the following parallel port control signals:

- Data strobe to peripheral equipment (parallel data bit 0)
- Automatic line feed (parallel data bit 1)
- Device initialization (parallel data bit 2)
- Device selection (parallel data bit 3)
- Parallel port interrupt enable (parallel data bit 4)

Figure 18-3 illustrates the 25-pin, D-type connector used in this computer. This connector provides Centronics-type signals for the parallel printer. Table 18-10 describes the signals used by this connector and lists their associated pin numbers.

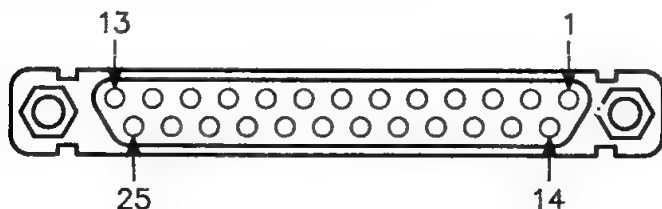


Figure 18-3. Parallel Connector

Serial and Parallel Communications

Table 18-10. Parallel Connector

PIN NUMBER	SIGNAL NAME	DESCRIPTION
1	<u>STROBE</u>	Strobe bit. This active-low pulse indicates that the computer is transmitting parallel data. It is used to time the data being sent to the peripheral device.
2	PDATA0 – PDATA7	Data bits 0 through 7. These signals are the system buffered and latched data bits.
10	<u>ACK</u>	Acknowledge. This active-low signal indicates that the peripheral device has received the data. It can be used for hardware handshaking.
11	BUSY	Busy. This signal indicates that the peripheral device is busy and not ready to receive data. It can be used for hardware handshaking.
12	PE	Paper end (out). This signal indicates a peripheral fault. It is used by a printer to indicate that it is out of paper.
13	SLCT	Select. This signal informs the peripheral that it has been selected.
14	<u>AUTO FDXT</u>	Automatic feed. This active-low signal requests a paper feed by the peripheral.
15	<u>ERROR</u>	Printer fault. This active-low signal indicates that an error condition exists in the peripheral.
16	<u>INIT</u>	Initialize. This active-low signal is used to initialize the peripheral.
17	<u>SLCT IN</u>	Select in. This active-low signal is used by the peripheral to select the computer.
18 – 25	GND	Ground.



ROM-Based Tests and Error Messages

This chapter describes the ROM-based self-tests, selectable tests, and the error messages you may encounter while using this system. Self-test routines initialize the major devices in the system when the computer is turned on. The selectable tests exercise specific circuits more intensely than the self-tests. In most cases, the selectable tests can be used to troubleshoot major circuits in the computer. An error message is a short line of text that is displayed if a test detects an error condition. The nature of the message is conclusive to the device or circuits that failed.

Self-Tests

On powerup the computer automatically executes a series of tests to verify that all of the circuits are in a starting configuration. These tests also check basic operational functions of the computer. The following systems are exercised during the self test sequence:

- Setup menu configuration
- Crystal frequencies
- Interrupt controllers
- DMA (direct memory access) controllers
- Disk drive controllers
- Disk drives (if enabled by Setup/Configuration Program)
- CPU
- ROM
- RAM.

ROM-Based Tests and Error Messages

Various other system-level controllers may be exercised during the self-tests. For example, video, keyboard, refresh, and any other controller in an expansion slot with an initialization ROM will be initialized during the self-test sequence. When the tests are finished, the computer will display an opening message or start the automatic boot procedure (autoboot). If autoboot to the floppy drive is started, a disk must be placed in the drive and the door shut within about 20 seconds.

Selectable Tests

The Monitor program provides a great deal of flexibility and power to the user. Among the commands available (most of which are described in Chapter 6) is the TEST command. Some of the tests that can be performed from this command are more comprehensive than the powerup self-tests and can help to quickly evaluate a number of apparent problems in memory, the disk drives, and the keyboard.

Test Menu

The ROM-based selectable tests should be run after the computer is first turned on and before any application or other programs are run. The reason for this is that some programs modify the interrupts that are used during the tests. This can produce unpredictable and deceptive results.

The tests are identified in the Monitor program's command summary as "Extended diagnostics." To display a menu from which the tests can be executed, type TEST at the Monitor program prompt and press the RETURN key. The menu is illustrated in Figure 19-1.

ROM-Based Tests and Error Messages

CHOOSE ONE OF THE FOLLOWING

1. DISK READ TEST
2. KEYBOARD TEST
3. BASE MEMORY TEST
4. EXPANSION MEMORY TEST
5. POWER-UP TEST
6. EXIT

ENTER YOUR CHOICE:

Figure 19-1. Test Menu

To execute a test, press the number that corresponds to the desired test. The program will immediately start executing that test.

The Disk Read Test

The disk read test continuously reads the first sector of track 0 on the default drive. This sector is the first sector of the boot track and must be read successfully for the computer to be able to boot a disk. By continuously reading the first boot sector, you can determine if you have a reliable disk/drive combination. Errors that occur during this test can be attributed to the disk, the drive, or both. If the test immediately displays an error message, use a known formatted disk before assuming you have a hardware-oriented error.

NOTE: Even though the boot track is being read, the disk does not need an operating system on it. Any formatted disk, whether it has the operating system installed on it or not, can be used for this test. On formatted data disks, sector 1 of track 0 contains a short routine that, when booted and executed, will display NO SYSTEM and halt the computer. This message indicates that the disk is formatted but does not contain an operating system.

The disk read test is non-destructive. This means that the operating system (MS-DOS) or data on the disk you are testing will not be erased or destroyed by the test. During operation, the disk read test will display the test count, indicating the number of times that the first track has been successfully read.

ROM-Based Tests and Error Messages

The Keyboard Test

The keyboard test allows you to check most of the keys on the keyboard and generate the key scan codes described in Chapter 8. Some of the keys may cause the computer to exit the test rather than produce a display.

This test displays several items of information. Each time a key that produces a printable ASCII character is pressed, the entire display will be filled with that character. The character scan code will be displayed in the upper right corner of the screen if the key produces one. If the key does not produce a code, the last code displayed will remain on the screen.

This test is also provides a quick way to fill the screen with data to check an external video monitor.

The Base Memory Test

During the power-up self-tests, only the first and last bank of base memory is tested. This test checks all base system memory and video memory for errors. While video memory is being tested, various patterns will appear on the screen.

The test produces a short click sound and displays a number indicating the current memory test count. This count is updated each time the test is successfully completed. The upper-right corner of the screen displays the current 64K bank of memory being tested in hexadecimal format. If an error is detected, the test stops and information about the error is displayed. The message reports the address where the error was detected, along with the bit number and integrated circuit (chip) number.

ROM-Based Tests and Error Messages

The Expansion Memory Test

The expansion memory test checks memory above the 1-megabyte boundary. The test produces a short click sound and displays a number indicating the current memory test count. This count is updated each time the test is successfully completed. The upper-right corner of the screen displays the current 64K bank of memory being tested in hexadecimal format. If an error is detected, the test stops and information about the error is displayed. The message reports the address where the error was detected, along with the bit number. An integrated circuit (chip) number is not displayed for expansion memory.

The Power-up Test

The power-up test repeatedly runs the self-tests described at the beginning of this chapter. These tests check all major circuits. While the powerup self-tests are not all-inclusive, some random fault conditions can be detected using this test. The disk-based diagnostics are more useful in exhaustively and repeatedly testing key elements of the computer. For more information on the disk-based diagnostics, refer to the end of this chapter.

Exiting the Test Menu

All ROM-based selectable tests will display a status screen while running. To exit any test, press the ESC key. With the exception of the keyboard test, the ESC key first halts the test. Pressing it a second time returns the program to the test menu. When you have exited to the menu, press the 6 to return to the Monitor program. The Monitor program prompt will be displayed on the screen.

NOTE: Do not use the CTRL-ALT-INS or CTRL-ALT-DEL key sequence to exit any of the tests. Some of the tests modify the interrupt vector table or the interrupts themselves. Exiting directly to the Monitor program can leave the computer in an unknown state, which often produces unpredictable results.

Error Messages

An error message may be generated whenever a test algorithm (self-test or selectable test) detects that a specific system, circuit, or component in the computer is not functioning as expected. The displayed message indicates the type of failure. This section presents the error messages in three categories:

- Error messages related to disk drives
- Error messages related to the Setup/Configuration program
- General error messages.

The error summary lines that may be displayed with certain error messages are explained at the end of this section.

Error Messages Related to Disk Drives

+++ DISK ERROR: Drive not ready! +++

This error message is displayed when the system attempts to boot an operating system but no disk has been placed in the floppy disk drive or the hard disk (if present and attempting to boot) has not been PREP'd or formatted. A faulty floppy disk or controller card can also cause this error message to be generated.

On floppy disk drive systems, make sure that a disk is correctly placed in the drive and that the drive latch is closed. Try to boot the system and notice if the error occurs again. If it does, you may need to check cable connections and the configuration jumpers on the floppy disk drive. Typical drive configuration information is presented in Chapter 4 of this manual.

On a hard disk drive system, try to determine if the problem exists in a piece of equipment other than the drive itself. Perform diagnostic tests (ROM-based or disk-based if necessary) on the hard disk controller card, and check for proper cabling and drive unit configuration. If no conclusive results are obtained from these checks, you may need to PREP and format the drive unit (assuming it has been done before).

ROM-Based Tests and Error Messages

+++ DISK ERROR: Bad disk controller' +++
+++ DISK ERROR: DMA Overrun' +++

These error messages usually indicate a fault in the disk controller card, but may be caused by addressing conflicts with other cards that are not supplied with this computer. It is also possible that the card is not seated properly or that cable connections are not correct.

+++ DISK ERROR: CRC error' +++
+++ DISK ERROR: Invalid address mark' +++
+++ DISK ERROR: Sector not found' +++

These error messages can happen when booting an operating system from a disk. They can also result from using an unformatted or defective disk, or from a faulty disk drive. Most often, this condition can be corrected by using another disk.

+++ DISK ERROR: Seek failure' +++

This error message indicates that the computer cannot read the disk or cannot read track 0. This error can be caused by a faulty disk, or by the failure of the disk controller card, or by a fault in the disk drive.

+++ DISK ERROR: Disk not bootable' +++

This error message occurs when the computer attempts to boot a disk that does not contain an operating system. The portion of the disk that normally contains the operating system may have been damaged.

+++ DISK ERROR: Invalid data read' +++

This error message indicates that the computer has read data from a disk but cannot interpret the data. The data on the disk may have lost its integrity due to exposure to an external magnetic field or due to physical damage of the disk itself. Try reading data from a known good disk and observe if the error is repeated.

ROM-Based Tests and Error Messages

+++ DISK ERROR: Data corrected! +++

This error message indicates that the drive has read data from a disk but the data was not all defined as either logic 0 or logic 1. However, the data was adequately defined to allow the controller to determine each data bit's state.

+++ DISK ERROR: Cannot reset drive! +++

This error message indicates a probable fault in the disk drive. If possible, substitute the suspect drive with a known good one to isolate the problem.

+++ DISK ERROR: Must run SETUP to boot from Winchester! +++

This error message indicates that a valid drive type must be specified in by the Setup/Configuration program. Refer to the configuration information in Chapter 5 for the correct procedure.

Error Messages Related to The Setup/Configuration Program

+++ ERROR: Please replace the backup battery! +++

This error message indicates that the real-time clock interface has lost power. This message will also appear upon the next powerup, after the backup battery has been replaced. If this message occurs and you do not have a replacement battery, press the ESC key to allow the system to boot.

+++ ERROR: Bad configuration information found in CMOS! +++

This message indicates that the CPU has found invalid data present in the special CMOS device. The Setup/Configuration program will be accessed automatically by the computer after this error is detected. At this time, system configuration data may be determined and stored in the CMOS device.

ROM-Based Tests and Error Messages

+++ ERROR: Base memory size error! SETUP: 512K ACTUAL: 640K +++

The CPU always checks the size of memory indicated in the Setup/Configuration program and compares it with the amount of actual memory installed in the computer. If the two figures are different, an error message similar to the one shown is generated. Be sure to update the Setup/Configuration program if additional memory is installed in your computer.

+++ ERROR: Expansion memory size error! SETUP: 15296K ACTUAL: 0K +++

An error message similar to the one shown is generated if the CPU detects a difference between the amount of expansion memory indicated by the Setup/Configuration program and the actual expansion memory installed. In the example, the actual memory is zero. This could mean that a major failure of memory devices has occurred or that the memory has been removed from the system.

General Error Messages

+++ ERROR: Memory parity failure! +++

+++ ERROR: System control processor failure! +++

These messages indicate that the system control processor on the I/O card is not responding to system requests. The keyboard and system will not function when this occurs.

+++ ERROR: CMOS memory failure! +++

This error message indicates that the CMOS RAM chip (real-time clock) cannot be properly read or written to.

+++ ERROR: CPU failure! +++

+++ ERROR: ROM checksum failure! +++

When any of these error messages occur, it indicates a fatal condition. Either the CPU is defective or the BIOS code in the ROM has been corrupted. In either case the device must be replaced.

ROM-Based Tests and Error Messages

+++ ERROR: Parity hardware failure! Address 00000.123D,
Chip 209/U228 +++

An error message similar to the one shown indicates that the CPU is unable to read or write to the system RAM. The routine will attempt to display a number to indicate which chip appears to have failed.

+++ ERROR: Keyboard not responding or not connected! +++

Normally this error message is caused by the keyboard being disconnected. If this is not the case, the problem may be in the keyboard controller IC, the keyboard cable, or on the computer's backplane board.

+++ ERROR: Timer interrupt failure! +++

This error message indicates possible failure of interrupt control or timer logic on the I/O card. Before replacing a card, make sure the card is properly seated and set up for the options installed. Also check that all optional cards are configured properly.

+++ ERROR: RAM failure! Address: 00000:123D, Bit n, Chip Uxxx +++

This error message will be generated if the system detects a faulty memory device in its environment. The address of the memory location, the specific bit number, and the chip number will be displayed.

+++ ERROR: Incorrect video configuration - Please run Setup! +++

Whenever this computer is powered up, a specific routine executes to determine what type of video interface is installed. The computer will default to that system even if the Setup/Configuration program is configured for some other type of video interface. This allows the display to be active at all times.

+++ Divide by zero! +++

This error message indicates an invalid mathematical operation and returns program execution to the Monitor program. The system may be booted using the boot command.

ROM-Based Tests and Error Messages

+++ Non-maskable interrupt! +++

The NMI error message indicates a possible program execution error or a power interruption. In either case, the system must be reset before it will continue to operate. Press the CTRL-ALT-DEL key sequence or if necessary turn the computer off, wait a few seconds, and then turn the computer back on.

+++ Overflow! +++

This error message occurs when the computer attempts to execute a mathematical calculation that exceeds the capabilities of the CPU. The system must be reset before it will continue to operate. Press the CTRL-ALT-DEL key sequence or if necessary turn the computer off, wait a few seconds, and then turn the computer back on.

+++ Wild interrupt! +++

This error message indicates that an interrupt generated by an unknown source was received. Program execution returns to the Monitor program. The system may be booted using the boot command.

+++ Wild hardware interrupt! +++

This error message occurs when a specific device generates an interrupt request that is out of sequence or unexpected. Program execution returns to the Monitor program. The system may be booted using the boot command.

+++ Cache circuit card failure! +++

This message indicates a cache circuit card failure at powerup or during the powerup test portion of the ROM-based tests.

Error Summary Lines

--- Errors found! Please press <ESC> to continue ---

When an error has been detected during the ROM-based tests, this error summary message will be displayed on the last line of the screen. Press the ESC key to clear the screen and continue the test.

ROM-Based Tests and Error Messages

--- Errors found! Please unlock keyboard, then press <ESC> to continue ---

This message may appear on the last line of the screen during ROM-based tests.. The keyboard must be unlocked before pressing the ESC key will be recognized. When the ESC is pressed, the screen will clear and the test will continue.

--- Fatal Error: Cannot Continue! ---

This message indicates that a major failure has occurred during a test. The test can not continue when a fatal error is encountered. This error is most commonly caused by defective RAM at address 0000H.

[Keyboard lock active: the keyboard is disabled]

This message is displayed just before the Monitor prompt or just before the system boots the operating system from a disk. It lets you know that the keyboard is disabled by the keyboard lock. The keyboard will not work until it is unlocked.

Disk-Based Diagnostics

The optional disk-based diagnostic package (Model CB-4164-41) is a supplemental testing package available for this computer. These tests contain comprehensive routines to check both the system core circuits and the interfaces required to operate peripheral devices. These tests can be very useful if you should encounter any difficulties with the computer.

The ROM-based tests check all functions of the hardware required to load and run the disk-based diagnostics. For a complete discussion on how to configure and execute disk-based diagnostics, refer to the disk-based diagnostic manual.



Appendix A

80386 Instruction Set

This appendix is an overview of the instruction set available for the 80386 microprocessor. This appendix does not contain technical details concerning the specific syntax or execution details of these instructions. Additional material may be obtained by referring to the related publications listed in Chapter 1 of this manual.

80386 Instruction Set

Data Transfer Instructions

General Purpose

MOV	Move operand
PUSH	Push operand onto stack
POP	Pop operand off stack
PUSHA	Push all registers onto stack
POPA	Pop all registers off stack
XCHG	Exchange Operand, Register
XLAT	Translate

Conversion

MOVZX	Move byte or Word, DWord, with zero extension
MOVSX	Move byte or Word, DWord, sign extended
CBW	Convert byte to Word, or Word to DWord
CDW	Convert word to DWord
CDWE	Convert word to DWord extended
CDQ	Convert DWord

Input/Output

IN	Input operand from I/O space
OUT	Output operand to I/O space

Address Object

LEA	Load effective address
LDS	Load pointer into D segment register
LES	Load pointer into E segment register
LFS	Load pointer into F segment register
LGS	Load pointer into G segment register
LSS	Load pointer into S(stack) segment register

Flag Manipulation

LAHF	Load A register from flags
SAHF	Store A register in flags
PUSHF	Push flags onto stack
POPF	Pop flags off stack
PUSHFD	Push Eflags onto stack
POPFD	Pop Eflags off stack
CLC	Clear carry flag
CLD	Clear direction flag
CMC	Compliment carry flag
STC	Set carry flag
STD	Set direction flag

Arithmetic Instructions**Addition**

ADD	Add operand
ADC	Add with carry
INC	Increment operand by 1
AAA	ASCII adjust for addition
DAA	Decimal adjust for addition

Subtraction

SUB	Subtract operand
SBB	Subtract with borrow
DEC	Decrement operand by 1
NEG	Negate operand
CMP	Compare operands
AAS	ASCII adjust for subtraction

Multiplication

MUL	Multiply double/signle precision
IMUL	Integer multiply
AAM	ASCII adjust after multiply

80386 Instruction Set

Division

DIV	Divide unsigned
IDIV	Integer divide
AAD	ASCII adjust after division

String Instructions

MOVS	Move byte or Word, DWord string
INS	Input string from I/O space
OUTS	Output string to I/O space
CMPS	Compare byte or Word, DWord string
SCAS	Scan byte or Word, DWord string
LODS	Load byte or Word, DWord string
STOS	Store byte or Word, DWord string
REP	Repeat
REPE/ REPZ	Repeat while equal/zero
RENE/ REPNZ	Repeat while not equal/not zero

Logical Instructions

Logicals

NOT	"NOT" operand
AND	"AND" operand
OR	"Inclusive OR" operand
XOR	"Exclusive OR" operand
TEST	"Test" operand

Shifts

SHL/SHR	Shift logical left/right
SAL/SAR	Shift arithmetic left/right
SHLD/SHRD	Double shift left/right

Rotates

ROL/ROR	Rotate left/right
RCL/RCR	Rotate through carry left/right

Bit Manipulation Instructions

Single Bit Instructions

BT	Bit test
BTS	Bit test and set
BTR	Bit test and reset
BTC	Bit test and compliment
BSF	Bit scan forward
BSR	Bit scan reverse

Bit String Instructions

IBTS	Insert bit string
XBTS	Exact bit string

Program Control Instructions

Conditional Transfers

SETCC	Set byte equal to condition code
JA/JNBE	Jump if above/not below nor equal
JAE/JNB	Jump if above or equal/not below
JB/JNAE	Jump if below/not above or equal
JBE/JNA	Jump if below or equal/not above
JC	Jump if carry
JE/JZ	Jump if equal/zero
JG/JNLE	Jump if greater/not less nor equal
JGE/JNL	Jump if greater or equal/not less
JL/JNGE	Jump if less/not greater nor equal
JLE/JNG	Jump if less or equal/not greater
JNC	Jump if not carry
JNE/JNZ	Jump if not equal/not zero
JNO	Jump if not overflow
JNP/JPO	Jump if not parity/parity odd
JNS	Jump if not sign
JO	Jump if overflow
JP/JPE	Jump if parity/parity even
JS	Jump if sign

80386 Instruction Set

Unconditional Transfers

CALL	Call procedure/task
RET	Return from procedure/task
JMP	Jump

Iteration Controls

LOOP	Loop
LOOPE/LOOPZ	Loop if equal/zero
LOOPNE/	
LOOPNZ	Loop if not equal/not zero
JCXZ	Jump if CX = 0

Interrupts

INT	Interrupt
INTO	Interrupt if overflow
IRET	Return from interrupt
CLI	Clear interrupt enable
STI	Set interrupt enable

High Level Language Instructions

BOUND	Check array bounds
ENTER	Setup parameter block for entering procedure
LEAVE	Leave procedure

Protection Model

SGDT	Store global descriptor table
SIDT	Store interrupt descriptor table
STR	Store task register
SLDT	Store local descriptor table
LGDT	Load global descriptor table
LIDT	Load interrupt descriptor table
LTR	Load task register
LLDT	Load local descriptor table
ARPL	Adjust requested privilege level
LAR	Load access rights
LSL	Load segment limit
VERR/VERW	Verify segment for reading or writing
LMSW	Load machine status word (lower 16 bits of CR0)
SMSW	Store machine status word

Processor Control Instructions

HLT	Halt
WAIT	Wait until busy# negated
ESC	Escape
LOCK	Lock bus





Appendix B

80287 Instruction Set

This appendix is an overview of the instruction set available for the 80287 coprocessor. This appendix does not contain technical details concerning the specific syntax or execution details of these instructions. Additional material may be obtained by referring to the related publications listed in Chapter 1 of this manual.

80287 Instruction Set

Data Transfer Instructions

FLD	Load
FST	Store
FSTP	Store and pop
FXCH	Exchange stack 1 and stack 0

Comparison Instructions

FCOM	Compare
FCOMP	Compare and pop
FCOMPP	Compare stack 1 to stack 0 and pop twice
FTST	Test stack 0
FXAM	Examine stack 0

Constant Instructions

FLDZ	Load +0.0 into stack 0
FLD1	Load +1.0 into stack 0
FLDPI	Load into stack 0
FLDL2T	Load log ₂ 10 into stack 0
FLDL2E	Load log ₂ e into stack 0
FLDLG2	Load log ₁₀ 2 into stack 0
FLDLN2	Load log _e 2 into stack 0

Arithmetic Instructions

FADD	Addition
FSUB	Subtraction
FMUL	Multiplication
FDIV	Division
FSQRT	Square root of stack 0
FSCALE	Scale stack 0 by stack 1
FPREM	Partial remainder of stack 0 + stack 1
FRNDINT	Round stack 0 to integer
FEXTRACT	Extract components of stack 0
FABS	Absolute value of stack 0
FCHS	Change sign of stack 0

Transcendental Instructions

FPTAN	Partial tangent of stack 0
FPATAN	Partial arctangent of stack 0 + stack 1
F2XM1	$2^{\text{stack 0}-1}$
FYL2X	$\text{Stack 1} * \log_2 \text{stack 0}$
FYL2XP1	$\text{Stack 1} * \log_2 (\text{stack 0} + 1)$

Processor Control Instructions

FINIT	Initialize NPX
FSETPM	Enter protected mode
FSTSWAX	Store status word
FLDCW	Load control word
FSTCW	Store control word
FSTSW	Store status word
FCLEX	Clear exceptions
FSTENV	Store environment
FLDENV	Load environment
FSAVE	Save state
FRSTOR	Restore state
FINCSTP	Increment stack pointer
FDECSTP	Decrement stack pointer
FFREE	Free stack 1
FNOP	No operation





Appendix C

80387 Instruction Set

This appendix is an overview of the instruction set available for the 80387 coprocessor. This appendix does not contain technical details concerning the specific syntax or execution details of these instructions. Additional material may be obtained by referring to the related publications listed in Chapter 1 of this manual.

80387 Instruction Set

Data Transfer

FLD	Load
FST	Store
FSTP	Store and pop
FXCH	Exchange

Comparison

FCOM	Compare
FCOMP	Compare and pop
FCOMPP	Compare and pop twice
FTST	Test stack 0
FUCOM	Unordered compare
FUCOMP	Unordered compare and pop
FUCOMPP	Unordered compare and pop twice
FXAM	Examine stack 0

Constants

FLDZ	Load +0.0 into stack 0
FLD1	Load +1.0 into stack 0
FLDPI	Load into stack 0
FLDL2T	Load log ₂ 10 into stack 0
FLDL2E	Load log ₂ e into stack 0
FLDLG2	Load log ₁₀ 2 into stack 0
FLDLN2	Load log _e 2 into stack 0

Arithmetic

FADD	Addition
FSUB	Subtraction
FMUL	Multiplication
FDIV	Division
FSQRT	Square root
FSCALE	Scale stack 0 by stack 1
FPREM	Partial remainder
FPREM1	Partial remainder (IEEE)
FRNDINT	Round stack 0 to integer
FXTRACT	Extract components of stack 0
FABS	Absolute value of stack 0
FCHS	Change sign of stack 0

Transcendental

FCOS	Cosine of stack 0
FPTAN	Partial tangent of stack 0
FPATAN	Partial arctangent
FSIN	Sine of stack 0
FSINCOS	Sine and cosine of stack 0
F2XM1	2stack 0-1
FYL2X	Stack 1 * log ₂ stack 0
FYL2XP1	Stack 1 * log ₂ (stack 0 + 1)

Processor Control

FINIT	Initialize NPX
FSTSWAX	Store status word
FLDCW	Load control word
FSTCW	Store control word
FSTSW	Store status word
FCLEX	Clear exceptions
FSTENV	Store environment
FLDENV	Load environment
FSAVE	Save state
FRSTOR	Restore state
FINCSTP	Increment stack pointer
FDECSTP	Decrement stack pointer
FFREE	Free stack 1
FNOP	No operation



Index

- 765, floppy disk controller
 - commands, 16-21
 - command bits, 16-23
 - drive status, 16-37
 - format a track, 16-31
 - interrupt status, 16-36
 - read data, 16-23
 - read deleted data, 16-25
 - read sector ID, 16-30
 - recalibrate, 16-36
 - scan equal, 16-32
 - scan high or equal, 16-35
 - scan low or equal, 16-33
 - seek, 16-38
 - specify, 16-37
 - write a track, 16-29
 - write data, 16-26
 - write deleted data, 16-28
 - device pinout, 16-15
 - digital output register, 16-6
 - initialization, 16-19
 - programming, 16-19
 - typical program steps, 16-19
 - output register
 - bit assignments, 16-6
 - registers
 - data, 16-10
 - bit assignments, 16-11
 - data transfer rates, 16-10
 - main status, 16-7
 - bit assignments, 16-8
 - floppy control
 - bit assignment, 16-10
 - serial controller, 16-4
- 6845 CRT controller
 - color select register, 17-28
 - data registers, 17-22
 - emulation, 17-20
 - palette definitions, 17-28
 - pointer address register, 17-21
 - port addresses, 17-21
 - port register, 17-26
 - register R8, 17-24, 25
 - resolution select, 17-27
 - status port register, 17-29

Index

- 80287 co-processor, 13-1
 - architecture, 13-2
 - bus interface unit, 13-3
 - condition codes, 13-4
 - control word, 13-6
 - device pinout, 13-11
 - numeric execution unit, 13-7
 - operation, 13-10
 - pin descriptions, 13-11
 - programming, 13-10
 - registers, 13-7
 - status word, 13-3
- 80386 CPU, 12-2
 - addressing modes, 12-17
 - architecture
 - base, 12-2
 - protected, 12-29
 - bus arbitration signals, 12-64
 - bus control signals, 12-65
 - bus cycle signals, 12-64
 - bus interface unit, 12-3
 - bus signals, 12-62
 - call gates, 12-34
 - co-processor interface signals, 12-66
 - description, 12-2
 - descriptors, 12-29
 - access rights byte definition, 12-44
 - attributes, 12-42
 - code, 12-44
 - data, 12-44
 - GDT, 12-29
 - IDT, 12-29
 - LDT, 12-29
 - system segment, 12-46
 - system segment types, 12-47
 - device pinout, 12-58
 - execution unit, 12-4
 - flag register, 12-9
 - description, 12-9
 - I/O, 12-19
 - I/O privilege, 12-32
 - instruction decode unit, 12-3
 - instruction pointer, 12-9
 - instruction prefetch unit, 12-3
 - inter-segment transfer, 12-33
 - interrupt processing, 12-20
 - interrupt signals, 12-66
 - interrupt types, 12-19

- interrupts, 12-19
 - priorities, 12-21
 - vector assignments, 12-21
- memory access, 12-31
- memory utilization, 12-18
- miscellaneous signals, 12-67
- modes
 - protected, 12-50
 - real, 12-28
 - virtual, 12-52
- page descriptor base register, 12-38
- page directory, 12-38
- page directory entries, 12-39
- page fault error codes, 12-41
- page table, 12-40
- page table entries, 12-39
- paging, 12-38
- paging unit, 12-4
- protected mode
 - initialization, 12-50
 - memory addressing, 12-49
- protection
 - privilege levels, 12-31
 - privilege types, 12-32
- protection concepts, 12-29
- real mode, 12-28
- registers
 - address, 12-14
 - control, 12-12
 - CR0 bit definitions, 12-13
 - debug, 12-16, 22
 - DR6 definition, 12-23
 - DR7 definition, 12-24
 - general purpose, 12-9
 - internal, 12-6
 - segment, 12-11
 - test, 12-16, 26
 - bit definitions, 12-27
 - TSS, 12-35
 - usage, 12-6
- segmentation unit, 12-4
- selector privilege, 12-32
- signal descriptions, 12-62
- status flags, 6-14
- system segment descriptors
 - gate, 12-48
 - LDT, 12-47
 - TSS, 12-47

Index

- task privilege, 12-32
- task switching, 12-35
- TLB, 12-40
- translation lookaside buffer, 12-40
- virtual mode, 12-52
 - addressing, 12-52
 - I/O permission, 12-53
 - interrupts, 12-55
 - paging, 12-52
 - protection, 12-53
 - transition, 12-56

- 80387 co-processor, 13-13
 - architecture, 13-13
 - bus control unit, 13-14
 - bus interface signals, 13-26
 - chip select signals, 13-27
 - condition codes, 13-17
 - control signals, 13-25
 - data interface/control unit, 13-15
 - device pinout, 13-23
 - floating point unit, 13-18
 - handshake signals, 13-25
 - operation, 13-21
 - pin descriptions, 13-25
 - programming, 13-22
 - quotient results, 13-18
 - registers, 13-19
 - status word, 13-15

- 8237 DMA controller, 14-36
 - block diagram, 14-36
 - mask command, 14-43
 - modes
 - block transfer, 14-46
 - cascade, 14-46
 - demand transfer, 14-46
 - single transfer, 14-45
 - operation, 14-44
 - pin descriptions, 14-49
 - pinout, 14-52
 - programming, 14-49
 - registers
 - base, 14-38
 - command, 14-39
 - current, 14-38
 - Internal, 14-38
 - mask, 14-42
 - mode, 14-41

- request, 14-43
- status, 14-39
- temporary, 14-40
- special features, 14-47

8254 interval timer, 7-3, 18

- description, 14-23
- modes
 - 0 — interrupt on terminal count, 14-26
 - 1 — programmable one-shot, 14-27
 - 2 — rate generator, 14-28
 - 3 — square wave rate generator, 14-29
 - 4 — software triggered strobe, 14-30
 - 5 — hardware triggered strobe, 14-31
- definitions, 14-25
- operation
 - read and write, 14-33
- pin descriptions, 14-34
- pinout, 14-35
- programming, 14-32
- read/write logic, 14-24
- registers
 - control, 14-24
- system timer interrupt, 14-3
- timer control, 14-24

8259 interrupt controller, 8-8, 14-1

- architecture, 14-3
- block diagram, 14-3
- cascading, 14-16
- commands
 - operation
 - word 2, 14-19
 - word 3, 14-20
 - initialization, 14-5
 - word 1, 14-6
 - word 2, 14-7
 - word 3, 14-7
 - word 4, 14-8
 - setup, 14-9
- Interrupt line assignments, 14-2
- Interrupt trigger, 14-15
- logic
 - control, 14-4
 - read/write, 14-4
- operation
 - CPU mode, 14-9
 - priority handling, 14-10
 - sequence, 14-17

Index

- pin descriptions, 14-21
 - pinout, 14-22
 - port address, 14-16
 - programming, 14-16
 - registers
 - in-service, 14-4
 - interrupt mask, 14-5
 - interrupt request, 14-4
 - resolver
 - priority, 14-4
 - status, 14-15
- 82C431 graphics controller, 17-58
- pin descriptions, 17-68
 - programming example, 17-60
 - register values, 11-26
- 82C432 sequencer, 17-73
- pin descriptions, 17-79
 - programming example, 17-74
 - register values, 11-26
- 82C433 attribute controller, 17-83
- pin descriptions, 17-90
 - programming example, 17-86
 - register values, 11-27
- 82C434 CRT controller, 17-95
- pin descriptions, 17-110
 - programming example, 17-101
 - register values, 11-27
- Attribute controller (see 82C433)
- Autoboot
- bypassing, 6-2
 - defeating, 6-2
- Breakpoints, 6-11
- cache memory description, 15-22
- CAS, 15-12
- computer
- autoboot, 2-8
 - boot procedure, 5-14
 - cache card, 4-9

- configuration
 - CPU card, 4-1
 - disk drives, 4-10
 - drive controller card, 4-9
 - drive modification, 4-10
 - I/O card, 4-4
 - memory card, 4-7
- features, 1-1
- general description, 1-1
- hardware
 - CPU card, 4-1
 - disk drives, 4-10
 - drive controller card, 4-9
 - drive modification, 4-10
 - I/O card, 4-4
 - memory card, 4-7
- Initialization, 6-2
- IC installation, 2-11, 12, 14
- internal options
 - CPU card, 4-1
 - disk drives, 4-10
 - drive controller card, 4-9
 - drive modification, 4-10
 - I/O card, 4-4
 - memory card, 4-7
- MFM-300, 2-9
- operating environment, 2-1
- power requirements, 2-1
- power-up, 2-8
- power-up sequence, 6-2
- reset, 2-10
- self test, 2-8
- setup, 2-2
- slot assignments, 2-13
- unpacking, 2-2
- CPU card
 - description, 12-1
- CRT controller (see 6845 and 82C434)
- Debugger, 6-8
- Disassembly
 - auxiliary fan, 3-3
 - backplane, 3-4
 - circuit card, 3-2
 - cover, 3-1
 - disk drive, 3-6
 - power supply, 3-5
- Disk-based diagnostics, 19-12

Index

Disk drives

- autoboot, 2-8, 6-2
- booting, 10-17
 - command bits, 16-23
 - commands, 16-21
 - configuration, 4-14, 18
- control register port address, 16-5
- controller, 16-3, 16-5
- DMA mode, 16-3
- drive selection, 16-2, 7
- drive status, 16-37
- drive types, 5-13
- error codes, 10-6
- error detection, 16-2
- floppy disks, 10-5
 - format track, 16-31
 - format, 10-10
- hard disk (*see* Hard disk drive)
- hard disk, 10-5
- identification codes
 - initialization, 10-8, 10
 - interrupt status, 16-36
- invalid commands, 16-2
 - manual commands, 2-9, 6-7
 - number of, 7-5
- parameters, 10-18, 21, 22, 24
- port 1F0H, 16-39
- port 1F1H, 16-40
- port 1F2H, 16-43
- port 1F3H, 16-43
- port 1F4H, 16-43
- port 1F5H, 16-43
- port 1F6H, 16-44
- port 1F7H, 16-44, 46
- port 3F2H, 16-6
- port 3F4H, 16-7
- port 3F5H, 16-10
- port 3F6H, 16-49
- port 3F7H, 16-10, 50
- read deleted data, 16-25
- read sector ID, 16-30
- read, 16-23
- recalibrate, 10-15, 16-36
- registers, 16-5
- ROM parameters, 10-18
 - scan equal, 16-32
 - scan high or equal, 16-35
 - scan low or equal, 16-33

- seek, 10-13
- seek, 16-38
- specify, 16-37
- supported, 16-1
- track
 - write deleted data, 16-28
 - write track, 16-29
 - write, 16-26
- Disk input/output
 - test drive ready, 10-14
- DMA controller (*see* 8237)
- EMS memory
 - description, 15-17
- Error messages
 - disk drive, 19-6
 - general, 19-9
 - setup/configuration, 19-8
 - summary lines, 19-11
- Expanded memory, 15-17
- Graphics controller (*see* 82C431)
- Hard disk drive
 - command register, 16-46
 - command register coding, 16-46
 - control registers, 16-39
 - controller commands, 16-51
 - cylinder high register, 16-43
 - cylinder low register, 16-43
 - data register, 16-39
 - diagnostic mode error codes, 16-40
 - digital input register, 16-50
 - digital input register codes, 16-50
 - drive and head register, 16-44
 - drive and head register codes, 16-44
 - error register, 16-40
 - fixed disk register codes, 16-49
 - format track command track layout, 16-54
 - operation mode error register codes, 16-40
 - port addresses, 16-39
 - programming, 16-51
 - sector count register, 16-43
 - sector interleave, 16-55
 - sector number register, 16-43
 - seek and restore step rate codes, 16-48
 - status register, 16-44
 - status register codes, 16-44
 - write precompensation register, 16-43

Index

Hardware

(Refer to specific device entries)

- 765 floppy disk controller
- 6845 CRT controller
- 80287 co-processor
- 80386 CPU
- 80387 co-processor
- 8237 DMA controller
- 8254 interval timer
- 8259 interrupt controller
- 82C431 graphics controller
- 82C432 sequencer
- 82C433 attribute controller
- 82C434 CRT controller
- MC146818A real time clock
- NS16450 asynchronous communications element
- SCP system control processor

I/O

- parallel connector, 2-7
- serial connector, 2-7

Installation

- tools, 2-1

INT 00H divide by zero, 7-1

INT 01H single step, 7-2

INT 02H non-maskable interrupt, 7-2

INT 03H software breakpoint, 7-2

INT 04H arithmetic overflow, 7-3

INT 05H print screen, 9-1

INT 08H timer, 7-3

INT 09H key pressed, 8-1

INT 0AH real-time clock, 7-3

INT 0BH communications (COM2:), 9-2

INT 0CH communications (COM1:), 9-2

INT 0DH parallel printer (LPT2:), 9-2

INT 0EH floppy disk interrupt return, 10-1

INT 0FH parallel printer (LPT1:), 9-2

INT 10H video input/output

functions

- 00H — set video mode, 11-2

- 01H — set cursor type, 11-3

- 02H — set cursor position, 11-3

- 03H — read cursor position, 11-4

- 04H — read light pen position, 11-4

- 05H — select active display page, 11-4

- 06H — scroll an area of the screen up, 11-4

- 07H — scroll an area of the screen down, 11-5

- 08H — read cursor position contents, 11-5

- 09H — send character and attribute to screen, 11-5

- 0AH — send character to screen, 11-5
- 0BH — set graphics foreground color, 11-5
- 0CH — write graphics pixel, 11-6
- 0DH — read graphics pixel, 11-6
- 0EH — dumb terminal display, 11-7
- 0FH — return video state, 11-7
- 10H — set palette registers, 11-7
- 11H — character generator routine, 11-8
 - BH register return values, 11-10
 - row specifier options, 11-9
- 12H — alternate select, 11-10
 - EGA information, 11-10
- 13H — write character string, 11-11
 - register, string, and cursor data, 11-11
- 64H — set scroll mode, 11-11
- 65H — set interlace mode, 11-12
- 66H — compatibility mode, 11-12
- function code summary, 11-2
- video modes, 11-2
- INT 11H equipment configuration, 7-4
 - summary, 7-4
- INT 12H memory size, 7-5
- INT 13H disk input/output
 - error status codes, 10-6
 - Functions
 - 00H — reset disk system, 10-7
 - 01H — read disk status, 10-8
 - 02H — read sector, 10-8
 - 03H — write sector, 10-9
 - 04H — verify sector, 10-9
 - 05H — format a track, 10-10
 - 08H — return drive parameters, 10-11
 - 09H — set hard drive parameters, 10-12
 - 0AH — read sectors with ECC bytes, 10-12
 - 0BH — write sectors with ECC bytes, 10-12
 - 0CH — seek, 10-13
 - 0DH — reset hard disk drive, 10-13
 - 0FH — return drive type, 10-13
 - 10H — test, 10-14
 - 11H — set drive type/recalibrate, 10-15
 - 12H — set media type, 10-15
 - 14H — hard disk controller self test, 10-16
 - 15H — return DASD type, 10-16
 - 18H — acknowledge cartridge changed, 10-17
 - 19H — reinitialize cartridge drive, 10-17
 - 1BH — assert recovery mode, 10-17
 - 1CH — deassert recovery mode, 10-17
 - Function codes, summary, 10-2

Index

INT 14H serial input/output, 9-2

Functions

- 00H — initialize serial input/output port, 9-3
- 01H — send character to the serial port, 9-4
- 02H — receive character from the serial port, 9-5
- 03H — read communications status, 9-5

line control status, 9-6

mode-select byte, 9-3

modem control status, 9-6

set baud rate, 9-4

set parity, 9-4

set word length, 9-3

INT 15H device control, 7-5

extended functions

- E2H — ram protect enable/disable, 7-12
- E3H — set processor speed, 7-13
- E4H — read processor speed, 7-13
- E5H — enable/disable cache memory, 7-14
- E6H — read cache state, 7-14

functions

- 80H — open device, 7-6
 - 81H — close device, 7-6
 - 82H — program terminate, 7-6
 - 83H — set non-busy wait, 7-7
 - 84H — read joystick, 7-7
 - 85H — system request key pressed, 7-7
 - 86H — busy wait interval, 7-7
 - 87H — block move, 7-8
 - 88H — extended memory size determination, 7-9
 - 89H — set processor to protected mode, 7-9
 - 90H — device busy loop entry point, 7-11
 - 91H — set interrupt complete flag, 7-12
- summary, 7-6

INT 16H keyboard input/output, 8-2

check keyboard buffer, 8-3

functions

- 00H — get character code, 8-3
 - 01H — check keyboard buffer, 8-3
 - 02H — get keyboard status, 8-3
 - 03H — set key repetition rate, 8-4
 - 05H — place character in buffer, 8-6
 - 10H — get extended character code, 8-6
 - 11H — check extended keyboard buffer, 8-6
 - 12H — get extended keyboard status, 8-6
- summary, 8-2

INT 17H printer input/output, 9-7

parallel printer status report, 9-7

printer function codes, 9-7

- INT 18H parallel/serial configuration, 9-8
 - handshake byte, 9-10
 - parallel map format, 9-9
 - parity/case byte, 9-11
 - serial map format, 9-10
 - word length, stop bits, parity, baud rate, 9-12
- INT 19H booting an operating system, 10-17
- INT 1AH set/read time of day, 7-14
 - functions
 - 00H — read time of day, 7-15
 - 01H — set time of day, 7-15
 - 02H — read real time clock, 7-15
 - 03H — set real time clock, 7-15
 - 04H — read real time clock date, 7-15
 - 05H — set real time clock date, 7-16
 - 06H — set real time clock alarm, 7-16
 - 07H — disable real time clock alarm, 7-16
 - summary, 7-14
- INT 1BH keyboard break, 8-7
 - key pressed, 8-1
 - keyboard break, 8-7
 - keyboard input/output, 8-2
 - booting an operating system, 10-17
 - disk drive
 - summary, 10-1
 - disk input/output, 10-2
 - disk parameters, 10-18, 22
 - floppy disk drive, 10-1
- INT 1CH tick timer, 7-17
- INT 1DH video initialization, 11-13
- INT 1EH disk parameters, 10-18
- INT 1FH defining characters, 11-14
- INT 40H disk input/output, 10-2
 - error status codes, 10-6
 - functions
 - 00H — reset disk system, 10-7
 - 01H — read disk status, 10-8
 - 02H — read sector, 10-8
 - 03H — write sector, 10-9
 - 04H — verify sector, 10-9
 - 05H — format a track, 10-10
 - 08H — return drive parameters, 10-11
 - 0CH — seek, 10-13
 - 0FH — return drive type, 10-13
 - 10H — test, 10-14
 - 11H — set drive type/recalibrate, 10-15
 - 12H — set media type, 10-15
 - 15H — return DASD type, 10-16
 - summary, 10-2

Index

- INT 4AH alarm interrupt, 7-17
- INT 70H real time clock alarm interrupt, 7-18
- INT 71H software redirect of IRQ2, 7-18
- INT 75H math coprocessor, 7-18
- INT 76H hard disk interrupt return, 10-1
- Interrupt controller (*see* 8259)
- Interrupt vectors
 - address description, 14-5
 - table, 14-5
- Interrupts (refer also to specific interrupt entry)
 - addresses, 6-19
 - alarm, 7-17
 - arithmetic overflow, 7-3
 - communications (COM1), 9-2
 - communications (COM2), 9-2
 - CPU, 7-1
 - default action, 6-19
 - defining characters, 11-14
 - device control, 7-5
 - divide by zero, 7-1
 - equipment configuration, 7-4
 - function codes, 9-3
 - hardware generated, 6-18
 - hardware, 6-17
 - get keyboard status, 8-6
 - INT (refer to specific interrupt entry)
 - maskable, 14-1
 - math co-processor, 7-18
 - memory size, 7-5
 - modifying, 6-19, 20
 - non-maskable, 14-1
 - non-maskable interrupt, 7-2
 - parallel printer (LPT2), 9-2
 - parallel printer, 9-2
 - parallel/serial configuration, 9-8
 - print screen, 9-1
 - printer input/output, 9-7
 - programming, 6-17
 - real time clock alarm, 7-18
 - real-time clock, 7-3
 - serial input/output, 9-2
 - set/read the time of day, 7-14
 - single step, 7-2
 - software breakpoint, 7-2
 - software redirect of IRQ2, 7-18
 - software, 6-19
 - summary, 6-21, 11-1, 9-1

- system summary, 7-1
- tick timer, 7-17
- time of day, 7-3
- timer, 7-3
 - using, 6-19
 - vector, 6-19
- video
 - initialization, 11-13
 - input/output, 11-1

Jumpers

- 1.2M drive address, 4-12
- 360K drive address, 4-11
- 720K drive address, 4-13
- configuration
 - CPU card, 4-3
 - I/O card, 4-6
- description
 - CPU card, 4-3

Keyboard

- adjustment, 5-8
- features, 5-1
- key descriptions
 - control, 8-21
 - function, 8-21
- key functions
 - alphanumeric, 5-2
 - alternate, 5-4
 - arrow, 5-5
 - backspace, 5-3
 - break, 5-7
 - caps lock, 5-2
 - control, 5-3
 - delete, 5-6
 - end, 5-5
 - enter/return, 5-3
 - escape, 5-7
 - function keys, 5-7
 - home, 5-5
 - insert, 5-6
 - number lock, 5-6
 - page down, 5-5
 - page up, 5-5
 - pause, 5-7
 - print screen, 5-7
 - scroll lock, 5-7
 - shift, 5-3

Index

- space bar, 5-3
 - system request, 5-7
 - tab, 5-3
 - keycodes
 - alphabetic, 8-9
 - control, 8-11
 - factored screen control, 8-12
 - function, 8-11
 - make/break, 8-26
 - numeric, 8-10, 15
 - punctuation, 8-10
 - repetition rates, 8-5
 - scan codes
 - alphabetic, 8-18
 - control, 8-24
 - function, 8-24
 - numeric, 8-20
 - punctuation, 8-20
 - status report, 8-4
 - AH register, 8-7
 - AL register, 8-7
 - XT/AT switch, 5-8
- Map**
- system memory, 6-22
 - port address, 6-23
- Maskable interrupts, 14-1**
- Mass storage**
- definition, 16-1
- MC146818A real time clock, 14-53**
- architecture, 14-54
 - internal memory, 14-64
 - interrupts, 14-62
 - alarm, 14-62
 - periodic, 14-62
 - update-enabled, 14-62
 - operation, 14-60
 - initialization, 14-61
 - update cycle, 14-63
 - periodic interrupt options, 14-57
 - pin descriptions, 14-65
 - pinout, 14-67
 - programming, 14-60
 - registers
 - alarm, 14-59
 - calendar, 14-59
 - internal, 14-55

- register A, 14-56
- register B, 14-57
- register C, 14-58
- register D, 14-59
- time, 14-59

Memory

- address decoding, 15-8
- addressing, 15-6
- bank error, 15-14
- board error, 15-14
- cache, 15-21
 - description, 15-22
 - read, 15-23
 - write, 15-26
- configuration, 15-15
- configuration options, 15-1
- cycles, 15-4
- data interface, 15-9
- EMS, 15-17
 - description, 15-17
- expansion, 15-1
- map, 6-22, 15-5
- pages, 15-3
- paging, 15-12
- parity
 - checking, 15-13
 - generation, 15-13
 - test, 15-15
- programming, 15-6
- read cycle, 15-4
- refresh, 15-12
- refresh cycle, 15-4
- search, 6-15
- slushware, 15-16
- system, 15-3
- types, 15-2
- user memory, 15-6
- write cycle, 15-4

Message

- ^ Invalid Command, 6-8
(see Error messages)

Monitor

- CRT
 - adjusting, 6-5

Monitor program

- basic features, 6-3
- command
 - color bar, 6-5
 - disk boot, 6-7
 - display memory, 6-9

Index

- examine memory, 6-10
 - examine/modify registers, 6-13
 - fill memory, 6-10
 - go (execute), 6-11
 - help, 6-5
 - hex math, 6-12
 - input from port, 6-12
 - move memory block, 6-13
 - output to port, 6-13
 - search memory, 6-15
 - set video/scroll mode, 6-6
 - trace user program, 6-15
 - unassemble, 6-16
 - command summary, 6-4
 - configuration, 5-9
 - description, 6-1
 - initial powerup sequence, 6-2
 - jump vectors, 6-24
 - setup, 5-9
 - setup/configuration
 - use, 5-9
 - video commands, 6-5
- Non-maskable interrupts, 14-1**
- NS16450, 18-2**
- baud rate/divisor latch values, 18-6
 - controller
 - interrupt, 18-3
 - modem, 18-3
 - handshaking, 18-4
 - input lines, 18-4
 - output lines, 18-4
 - Interrupts, 18-7
 - pin functions, 18-12
 - programmable options, 18-4
 - programming, 18-4
 - receiver, 18-2
 - register ports, 18-5
 - registers, 18-5
 - buffer receiver, 18-5
 - divisor latches, 18-5
 - interrupt, 18-6
 - line control, 18-8
 - line status, 18-10
 - line status report, 18-10
 - modem control, 18-9
 - modem status, 18-11
 - modem status report, 18-11
 - transmitter holding, 18-5
 - transmitter, 18-2

Operating system

- loading, 2-10

Parallel connector

- pinout, 18-17

Parallel port, 18-16

- addresses, 18-16

Peripherals

- connecting, 2-3

Port

- interrupt controller, 14-16

Product information, 1-5

- environment, 1-6

- basic unit, 1-6

- peripherals, 1-7

- performance options, 1-7

- co-processors, 1-7

- mass storage options, 1-8

- video options, 1-8

- specifications, 1-9, 10, 11

Print screen

- status byte, 9-1

Programmable interval timer (*see* 8254)**Program**

- execution

- halting, 6-11

Programming

- 8237, 14-49

- 8254, 14-32

- 8259, 14-16

- MC146818A, 14-60

- SCP, 14-79

RAS, 5-12**Raster-scan, 17-1****Real-time clock (*see* MC146818A)****Related publications, 1-3****ROM-based tests, 19-2**

- base memory, 19-4

- disk read, 19-3

- exiting, 19-5

- expansion memory, 19-5

- keyboard, 19-4

- powerup, 19-5

- test menu, 19-2

Tools

- required, 2-1

Index

SCP (system control processor), 14-67

buffers

input, 14-69

output, 14-69

commands, 14-70

command byte description, 14-72

keyboard

operations, 14-73

receiving data, 14-73

sending data, 14-74

operation

CPU protected mode, 14-75

pin descriptions, 14-76

programming

fast CPU reset, 14-79

fast gate A20, 14-79

NMI enable, 14-80

scratchpad RAM enable, 14-80

registers

status, 14-68

slushware, 14-74

Self-tests, 19-1

Serial connector

pinout, 18-2

Serial port

initializing, 9-3

Sequencer (*see* 82C432)

Slushware, 15-16

Software

video drivers, 4-24

Sound

programming, 7-18

System control processor (*see* SCP)

System memory, 15-3 (*see* Memory)

System memory map, 6-22

Video

9-pin connector

pinout, 2-4, 5

15-pin connector

pinout, 2-5

31 kHz, 17-118

color palette, 17-118

6845

color select register, 17-28

data registers, 17-22

light pen register, 17-30

- palette definitions, 17-28
- pointer address register, 17-21
- port addresses, 17-21
- port register, 17-26
- register R8, 17-24, 25
- resolution select, 17-27
- status port register, 17-29
- 82C431, 17-58
 - register values, 11-26
- 82C432, 17-73
 - character map A, 17-77
 - character map B, 17-78
 - clocking mode register, 17-75
 - memory mode register, 17-78
 - register values, 11-26
 - sequencer addressing, 17-73
 - sequencer register, 17-73
- 82C433, 17-83
 - addressing, 17-85
 - mode control register, 17-87
 - overscan register output, 17-87
 - palette register output, 17-87
 - register summary, 17-84
 - register values, 11-27
 - status port, 17-84
- 82C434, 17-95
 - addressing, 17-100
 - mode control register, 17-108
 - overflow register, 17-104
 - register summary, 17-99
 - register values, 11-27
- CGA, 17-37
 - color register select, 17-45
 - CRT control registers, 17-37
 - I/O port assignments, 17-43
 - I/O port selection, 17-44
 - palette 1 selection, 17-46
 - palette 2 selection, 17-46
 - port 3D8, 17-47
 - port 3DA, 17-48
 - programming, 17-49
 - status select, 17-49
- color attribute byte, 11-20, 17-12
- color modes, 11-16
- color selection, 11-21, 17-13
- configuration, 2-3, 4-21
 - options, 4-23

Index

- controller
 - 6845 emulation, 17-20
 - introduction, 17-7
- EGA, 11-15, 17-56
 - bit/memory relation, 17-61
 - color compare register, 17-62
 - external register values, 11-25
 - graphics controller addressing, 17-60
 - graphics controller register, 17-59
 - graphics register R3, 17-63
 - miscellaneous register, 17-66
- EGA modes, 11-18
- initialization
 - default values, 11-13
- emulation
 - CGA, 17-37
 - HGC, 17-50
- hardware features, 17-9
- HGC, 17-50
 - attributes, 17-55
 - configuration register, 17-54
 - data register, 17-50
 - display mode control register, 17-51
 - display status register, 17-52
 - index register, 17-50
- Initialization
 - mode set, 7-5
- MDA, 17-30
 - attribute codes, 17-33
 - character/attribute addressing, 17-32
 - control register, 17-30
 - CRTC registers, 17-37
 - register addresses, 17-36
 - status register, 17-36
- memory, 17-8
- modes, 6-6, 11-2, 18
 - basic, 17-10
 - CGA, 17-37

- color
 - high resolution, 11-22
 - medium resolution, 11-21
 - color selection, 11-21
 - palette, 11-22
- EGA, 17-56
- graphics, 11-21, 17-13
- HGC, 17-50
- mode at powerup, 7-5
- scroll modes, 6-6
- mode F attributes, 17-14
- text, 11-19, 17-10
- mode 10, 11-24
 - base colors, 11-24, 17-16
 - memory plane assignment, 11-24, 17-15
 - Palette color attributes, 11-25
- mode F, 11-22
 - attributes, 11-23
- monitor connection, 2-6
- monochrome attribute byte, 11-20, 17-12
- monochrome modes, 11-16
- operating modes, 11-18, 17-64
- overview, 17-1
- palette colors, 11-6, 22, 17-14
- palette register color attributes, 17-16
- raster scan, 17-1
- shift registers, 17-9
- systems, 17-17
 - 31 kHz, 17-19
 - CGA, 17-18
 - EGA, 17-18
 - HGC, 17-17
 - MDA, 17-17
 - PGC, 17-19
 - VGA, 17-19
- video encoder, 17-9
- Video emulation
 - MDA, 17-30
- Video sequencer (*see* 82C432)

